US007627725B2

(12) **United States Patent**
**Yoshida**

(10) **Patent No.: US 7,627,725 B2**
(45) **Date of Patent: Dec. 1, 2009**

(54) **STORED DATA PROCESSING APPARATUS, STORAGE APPARATUS, AND STORED DATA PROCESSING PROGRAM**

(75) Inventor: **Osamu Yoshida**, Kawasaki (JP)

(73) Assignee: **Fujitsu Limited**, Kawasaki (JP)

( * ) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 430 days.

(21) Appl. No.: **11/650,724**

(22) Filed: **Jan. 8, 2007**

(65) **Prior Publication Data**

US 2008/0077753 A1 Mar. 27, 2008

(30) **Foreign Application Priority Data**

Sep. 22, 2006 (JP) ..... 2006-256752

(51) **Int. Cl.**
***G06F 12/00*** (2006.01)

(52) **U.S. Cl.** ..... **711/156**; 711/100; 711/111; 711/112; 711/154

(58) **Field of Classification Search** ..... 711/100, 711/111–112, 114, 154, 156
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,515,500 A 5/1996 Mizuno et al.
2003/0218816 A1 11/2003 Katoh et al.
2006/0224841 A1 * 10/2006 Terai et al. ..... 711/154

FOREIGN PATENT DOCUMENTS

| | | |
|---|---|---|
| JP | 05-035416 | 2/1993 |
| JP | 6-332623 | 12/1994 |
| JP | 08-124318 | 5/1996 |
| JP | 11-086465 | 3/1999 |
| JP | 2000-112776 | 4/2000 |
| JP | 2003-346432 | 12/2003 |

* cited by examiner

*Primary Examiner*—Jasmine Song
(74) *Attorney, Agent, or Firm*—Greer, Burns & Crain, Ltd.

(57) **ABSTRACT**

An apparatus for enhancing error recovery capability while suppressing increases in response time for a higher-level device, including a management information storage section storing, for each second block, management information relating to a storage medium and information indicating whether a third block corresponding to the second block is valid or invalid; a calculation section performing calculation for each bit position using data of all first blocks in an update target in the case where the management information stored in the management information storage section indicates that the third block corresponding to the second block specified as the update area is invalid and outputs a result of the calculation as calculation data; and a write section writing the calculation data output from the calculation section in the third block associated with the update target and validates the third block in the management information stored in the management information storage section.

**18 Claims, 20 Drawing Sheets**

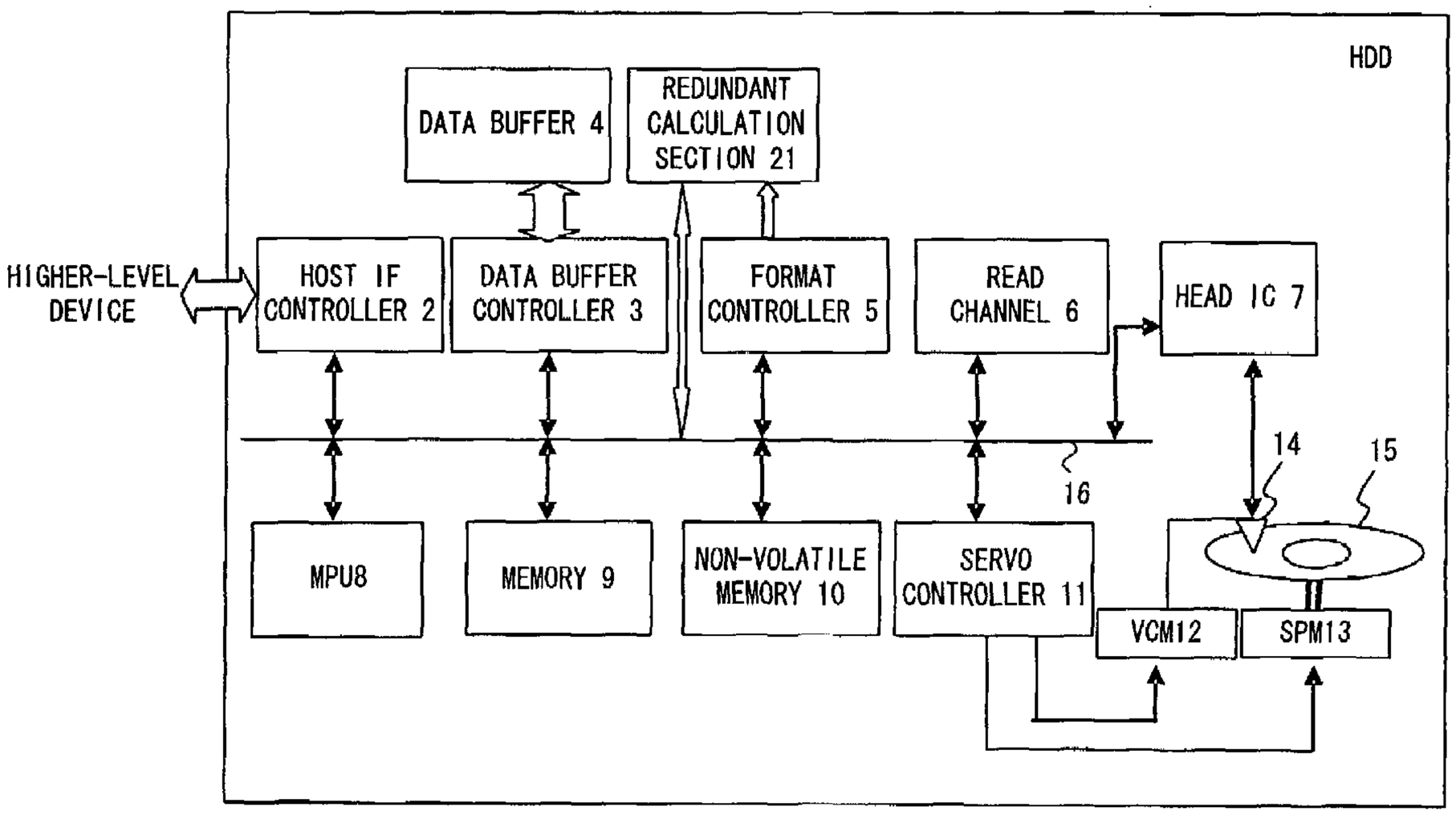


FIG. 1

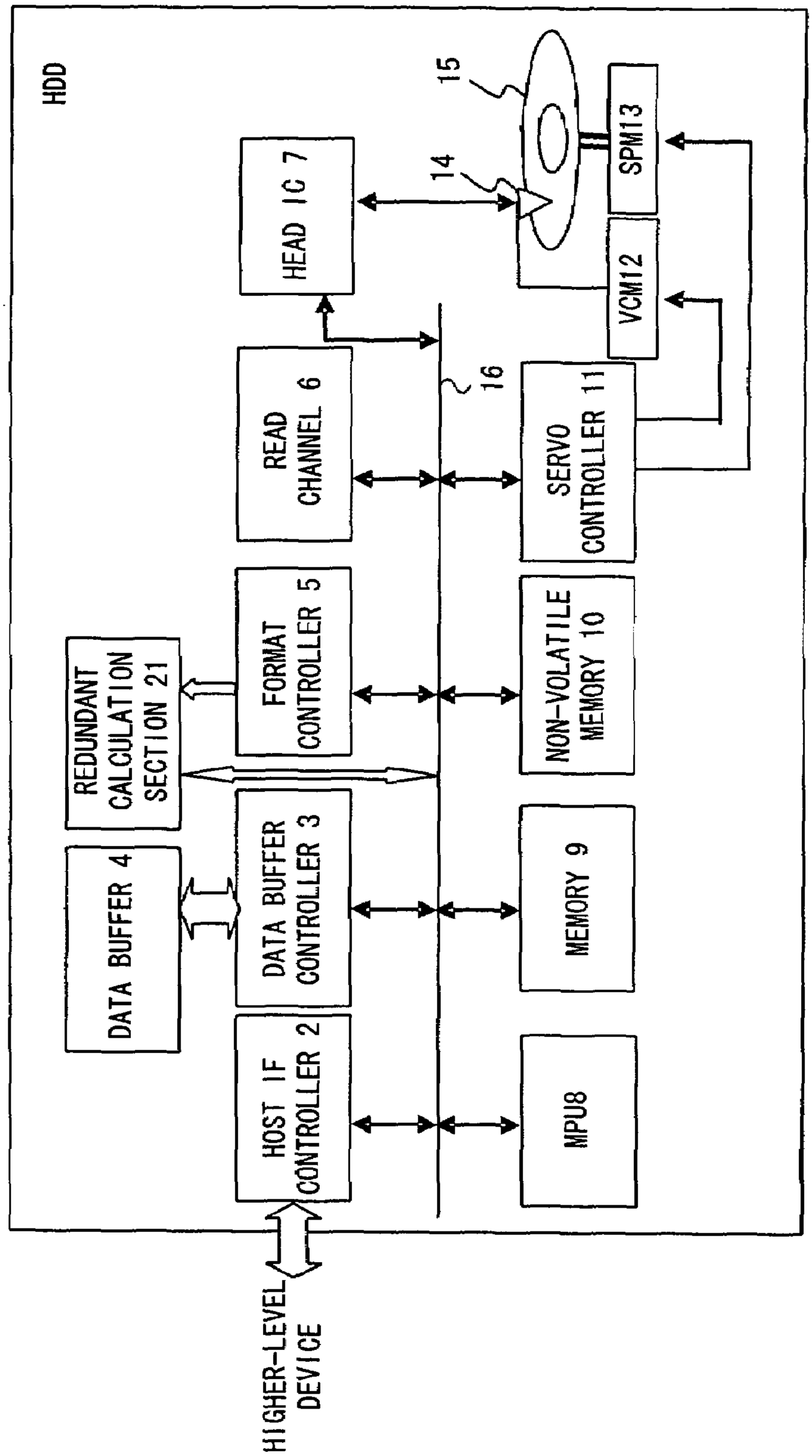

HDD
15
14
HEAD IC 7
SPM13
VCM12
16
READ CHANNEL 6
SERVO CONTROLLER 11
FORMAT CONTROLLER 5
NON-VOLATILE MEMORY 10
REDUNDANT CALCULATION SECTION 21
DATA BUFFER 4
DATA BUFFER CONTROLLER 3
MEMORY 9
HOST IF CONTROLLER 2
MPU8
HIGHER-LEVEL DEVICE

FIG. 2

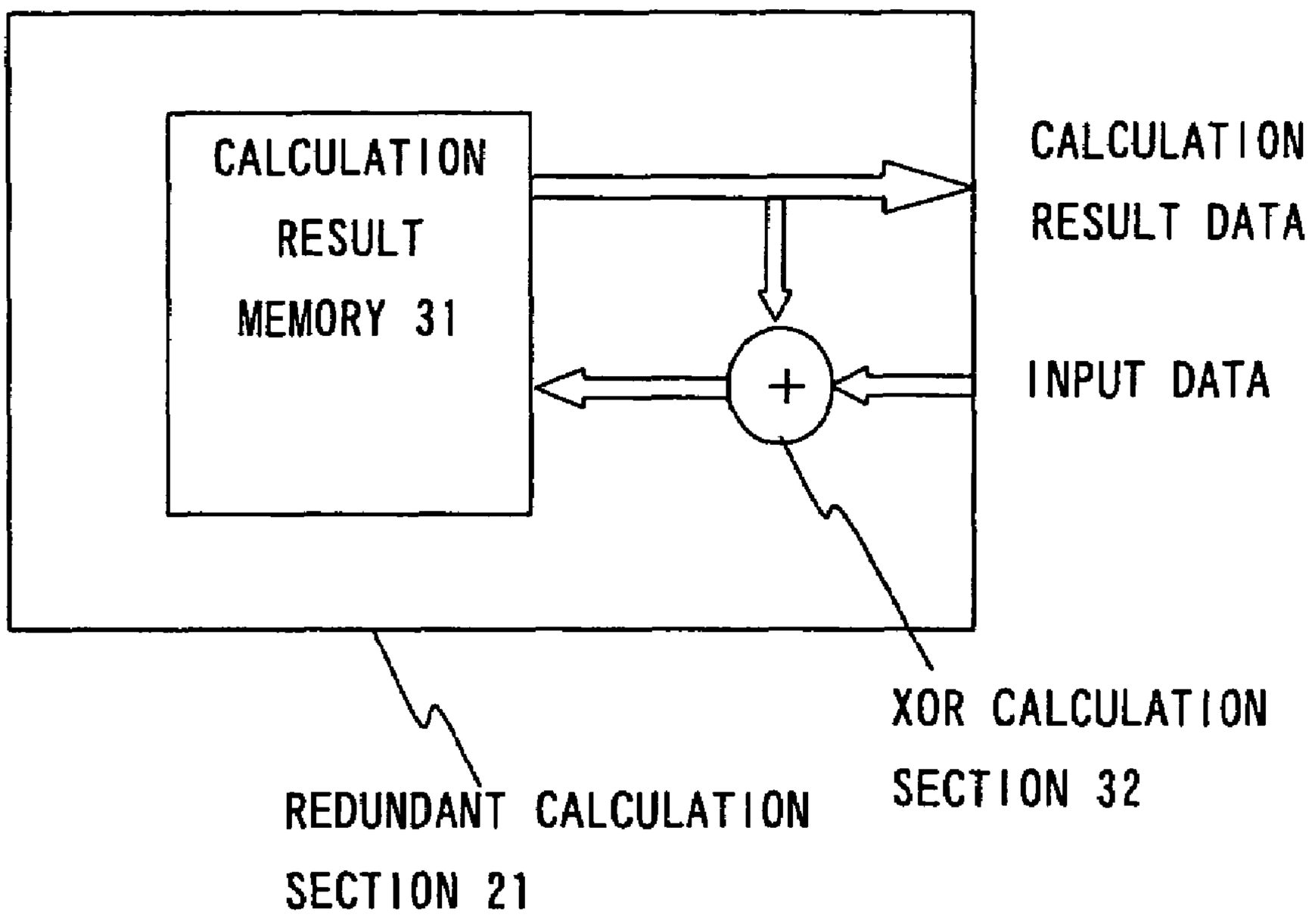

CALCULATION
RESULT
MEMORY 31
+
CALCULATION
RESULT DATA
INPUT DATA
XOR CALCULATION
SECTION 32
REDUNDANT CALCULATION
SECTION 21

FIG. 3

| | | | | | | |
|---|---|---|---|---|---|---|
| SEGMENT 0 | LBA 0 | LBA 1 | ... | | LBA m-1 | REDUNDANT 0 |
| SEGMENT 1 | LBA m | LBA m+1 | ... | | LBA 2m-1 | REDUNDANT 1 |
| SEGMENT 2 | LBA 2m | LBA 2m+1 | ... | | LBA 3m-1 | REDUNDANT 2 |
| : | | | | | | : |
| : | | | | | | : |
| SEGMENT n-1 | | ... | LBA LAST | | | REDUNDANT n-1 |

FIG. 4

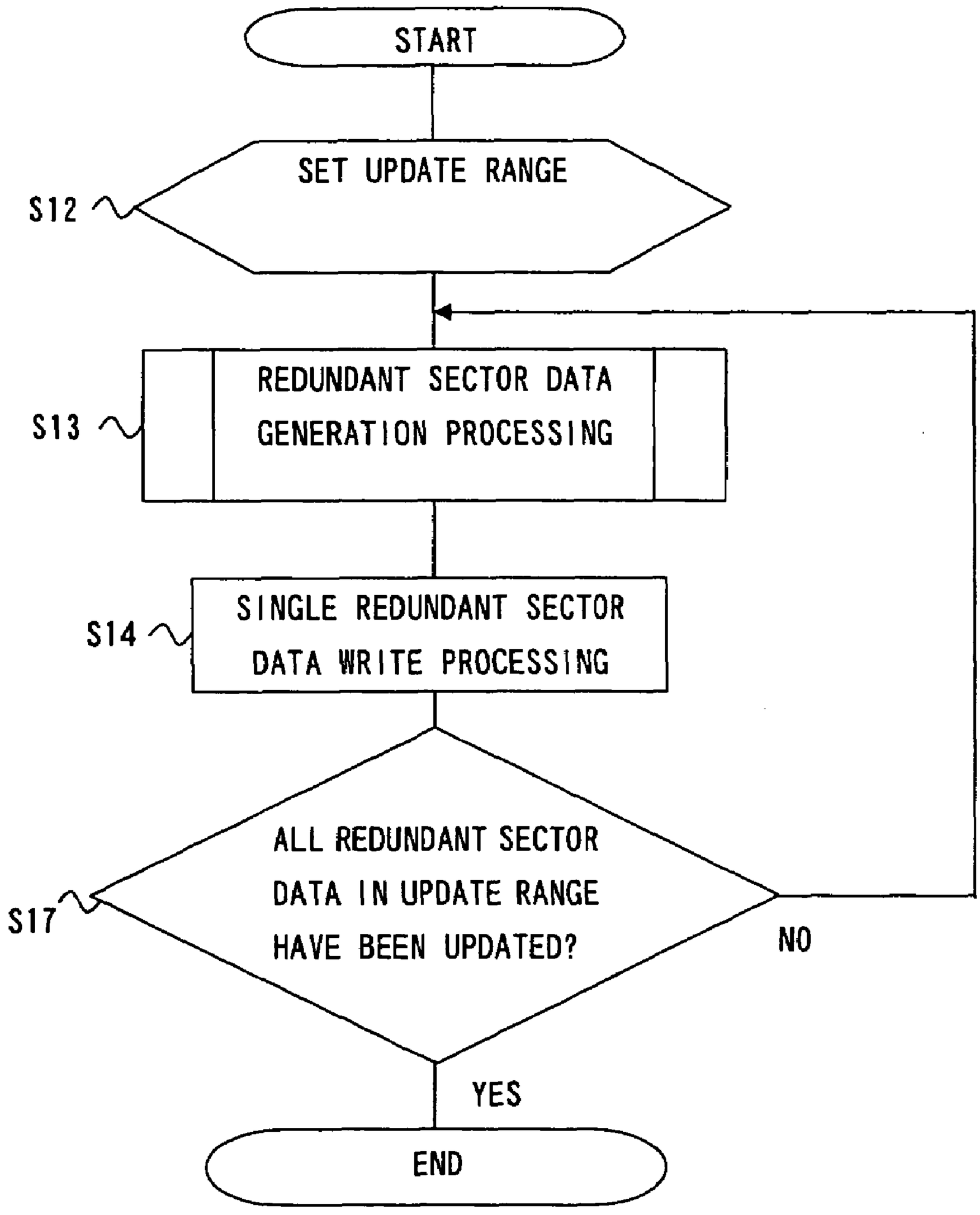

START
SET UPDATE RANGE
S12
REDUNDANT SECTOR DATA GENERATION PROCESSING
S13
SINGLE REDUNDANT SECTOR DATA WRITE PROCESSING
S14
ALL REDUNDANT SECTOR DATA IN UPDATE RANGE HAVE BEEN UPDATED?
S17
NO
YES
END

FIG. 5

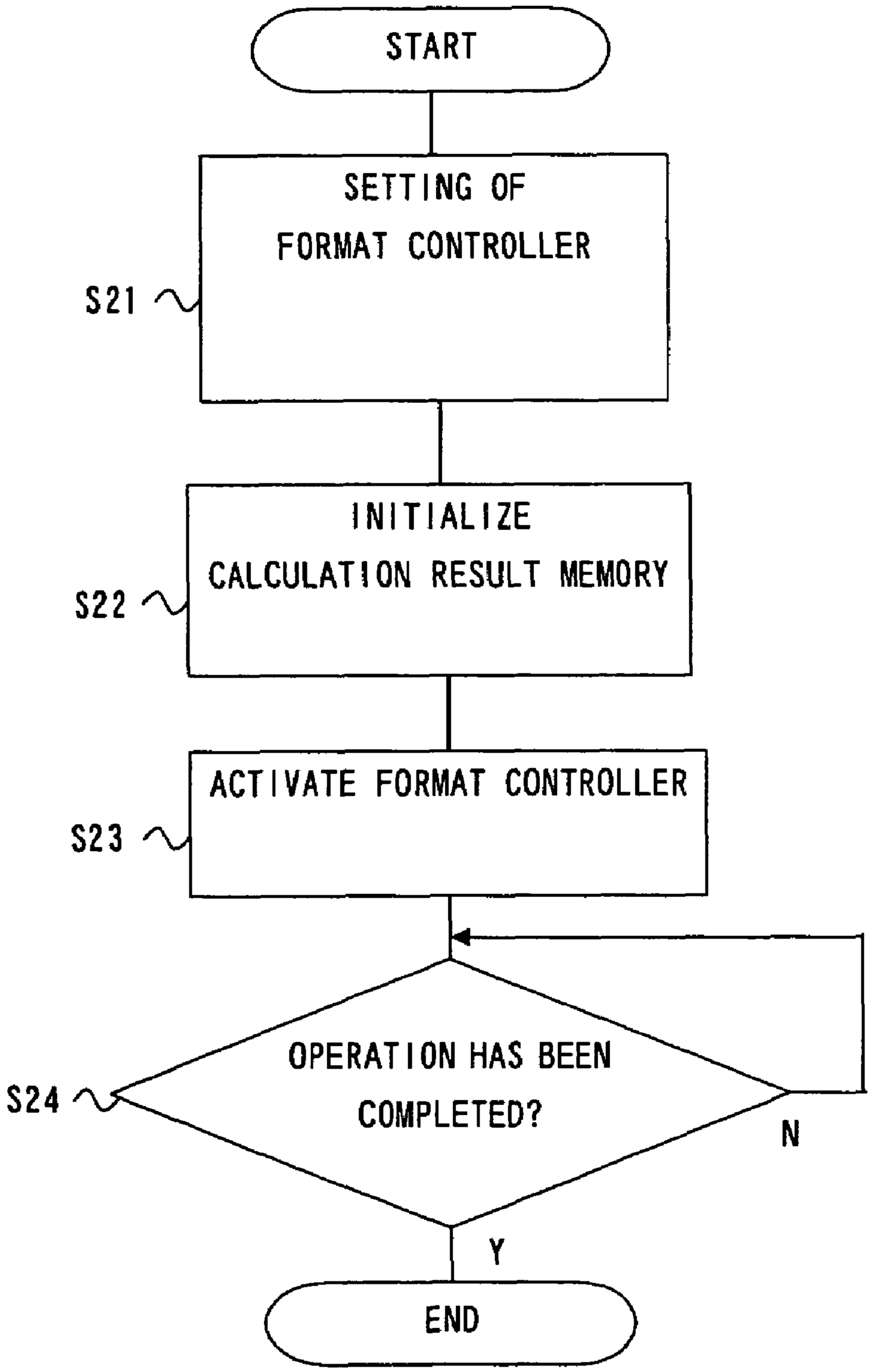

START
SETTING OF FORMAT CONTROLLER
S21
INITIALIZE CALCULATION RESULT MEMORY
S22
ACTIVATE FORMAT CONTROLLER
S23
OPERATION HAS BEEN COMPLETED?
S24
N
Y
END

FIG. 6

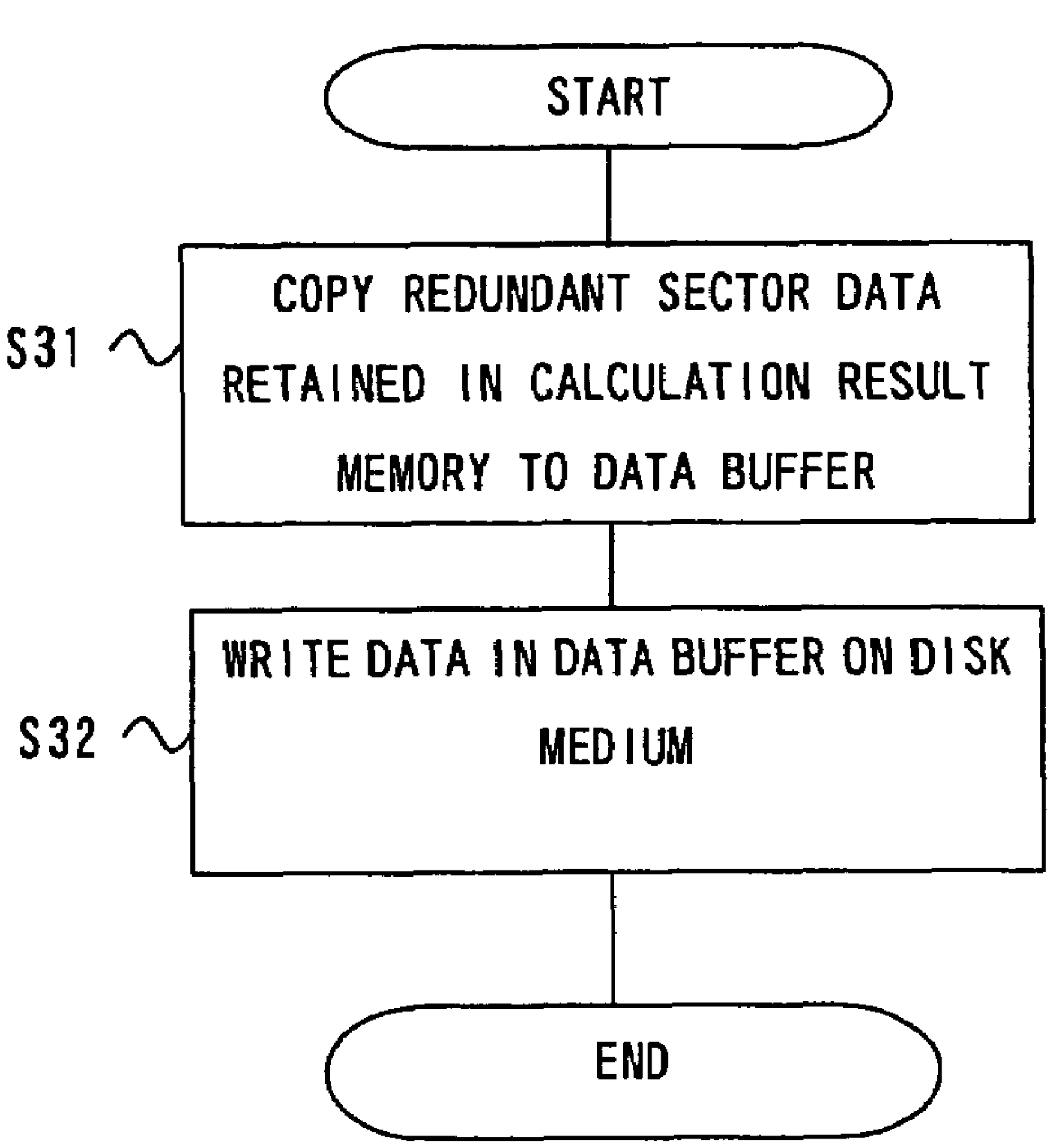

START
S31
COPY REDUNDANT SECTOR DATA RETAINED IN CALCULATION RESULT MEMORY TO DATA BUFFER
S32
WRITE DATA IN DATA BUFFER ON DISK MEDIUM
END

FIG. 7

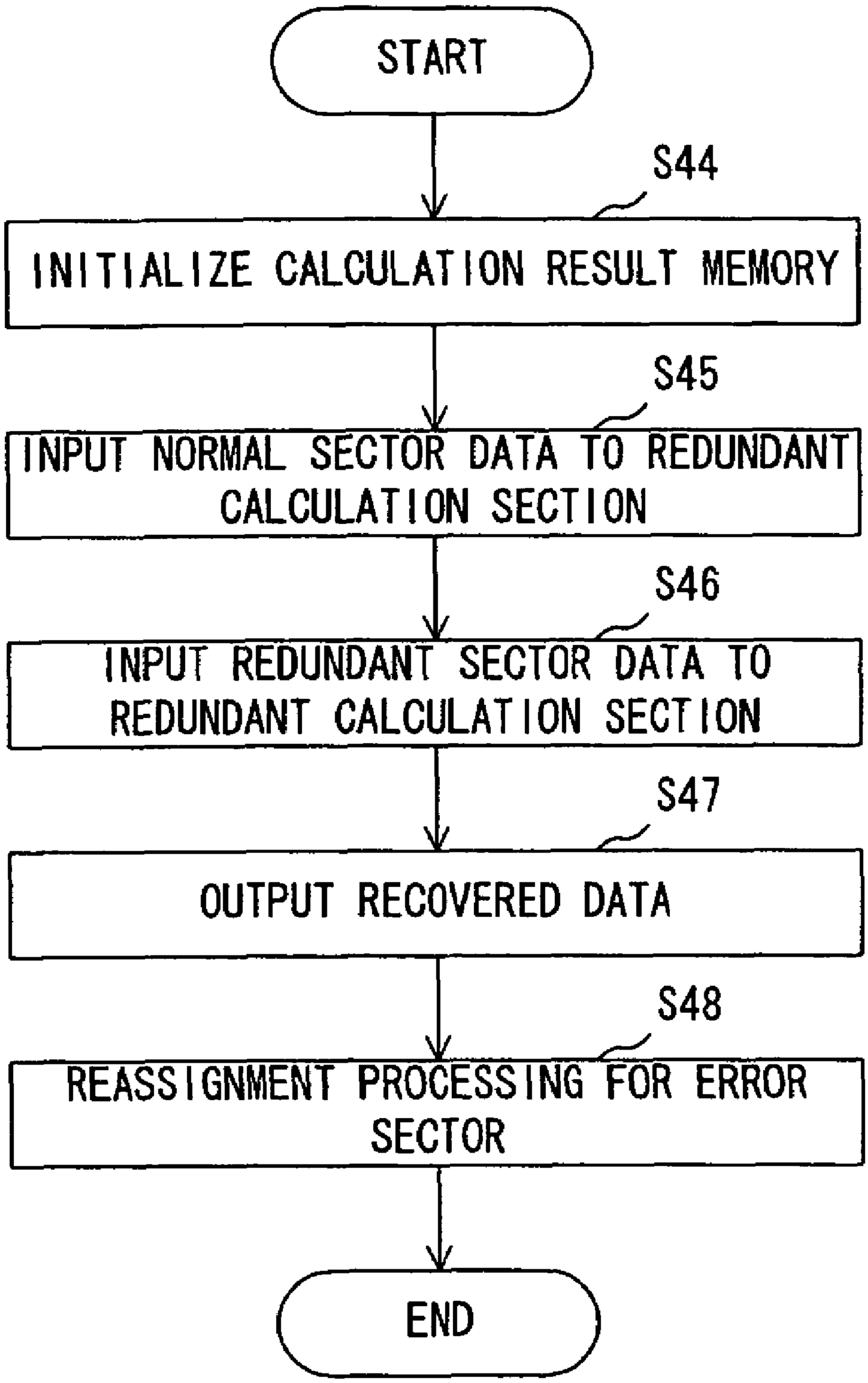

START
S44
INITIALIZE CALCULATION RESULT MEMORY
S45
INPUT NORMAL SECTOR DATA TO REDUNDANT CALCULATION SECTION
S46
INPUT REDUNDANT SECTOR DATA TO REDUNDANT CALCULATION SECTION
S47
OUTPUT RECOVERED DATA
S48
REASSIGNMENT PROCESSING FOR ERROR SECTOR
END

FIG. 8

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| SEGMENT 0 | LBA 0 | LBA 1 | ... | | ... | LBA m-2 | LBA m-1 |
| SEGMENT 1 | LBA m | LBA m+1 | ... | | ... | LBA 2m-2 | LBA 2m-1 |
| : : | | | | | | | : : |
| SEGMENT n-1 | | | | ... | LBA LAST | REDUNDANT 0 | REDUNDANT 1 |
| | REDUNDANT 2 | ... | | | ... | REDUNDANT n-2 | REDUNDANT n-1 |

FIG. 9

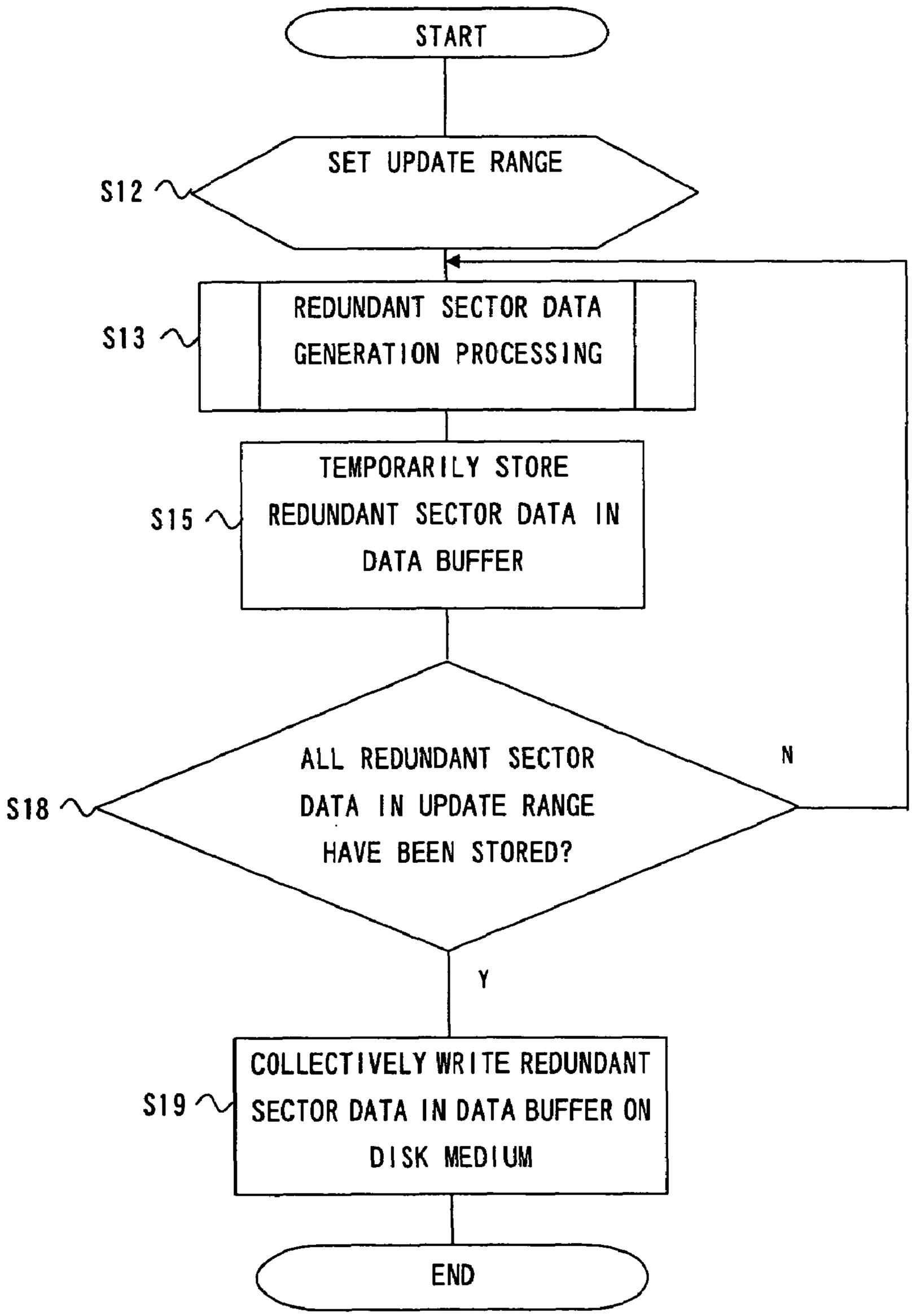

START
SET UPDATE RANGE
S12
REDUNDANT SECTOR DATA GENERATION PROCESSING
S13
TEMPORARILY STORE REDUNDANT SECTOR DATA IN DATA BUFFER
S15
ALL REDUNDANT SECTOR DATA IN UPDATE RANGE HAVE BEEN STORED?
S18
N
Y
COLLECTIVELY WRITE REDUNDANT SECTOR DATA IN DATA BUFFER ON DISK MEDIUM
S19
END

FIG. 10

| SEGMENT 0 | LBA 0 | LBA m+1 | LBA 2 | LBA m+3 | ... | ... | LBA m-2 | LBA 2m-1 | REDUNDANT 0 |
|---|---|---|---|---|---|---|---|---|---|
| SEGMENT 1 | LBA m | LBA 1 | LBA m+2 | LBA 3 | ... | ... | LBA 2m-2 | LBA m-1 | REDUNDANT 1 |

FIG. 11

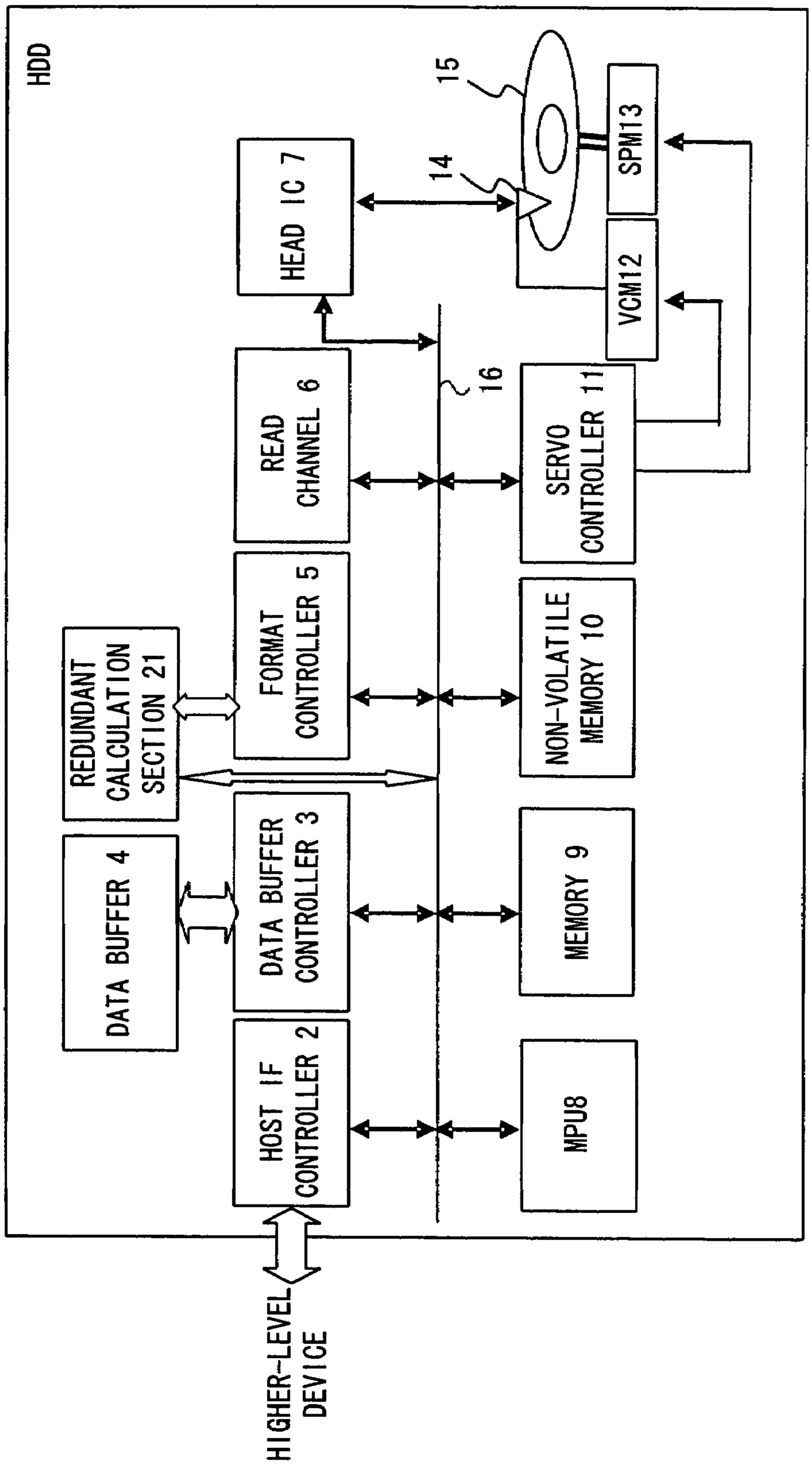

HDD
15
14
HEAD IC 7
SPM13
VCM12
16
READ CHANNEL 6
SERVO CONTROLLER 11
REDUNDANT CALCULATION SECTION 21
FORMAT CONTROLLER 5
NON-VOLATILE MEMORY 10
DATA BUFFER 4
DATA BUFFER CONTROLLER 3
MEMORY 9
HOST IF CONTROLLER 2
MPU8
HIGHER-LEVEL DEVICE

FIG. 12


START
S33
MAKE SETTING OF DATA PATH FOR FORMAT CONTROLLER SO AS TO ALLOW DATA TO BE DIRECTLY WRITTEN FROM REDUNDANT CALCULATION SECTION ONTO THE DISK MEDIUM
S34
WRITE REDUNDANT SECTOR DATA ONTO DISK MEDIUM IN THE SAME MANNER AS NORMAL PROCEDURE IN WHICH DATA ARE WRITTEN ONTO DISK MEDIUM
END

FIG. 13

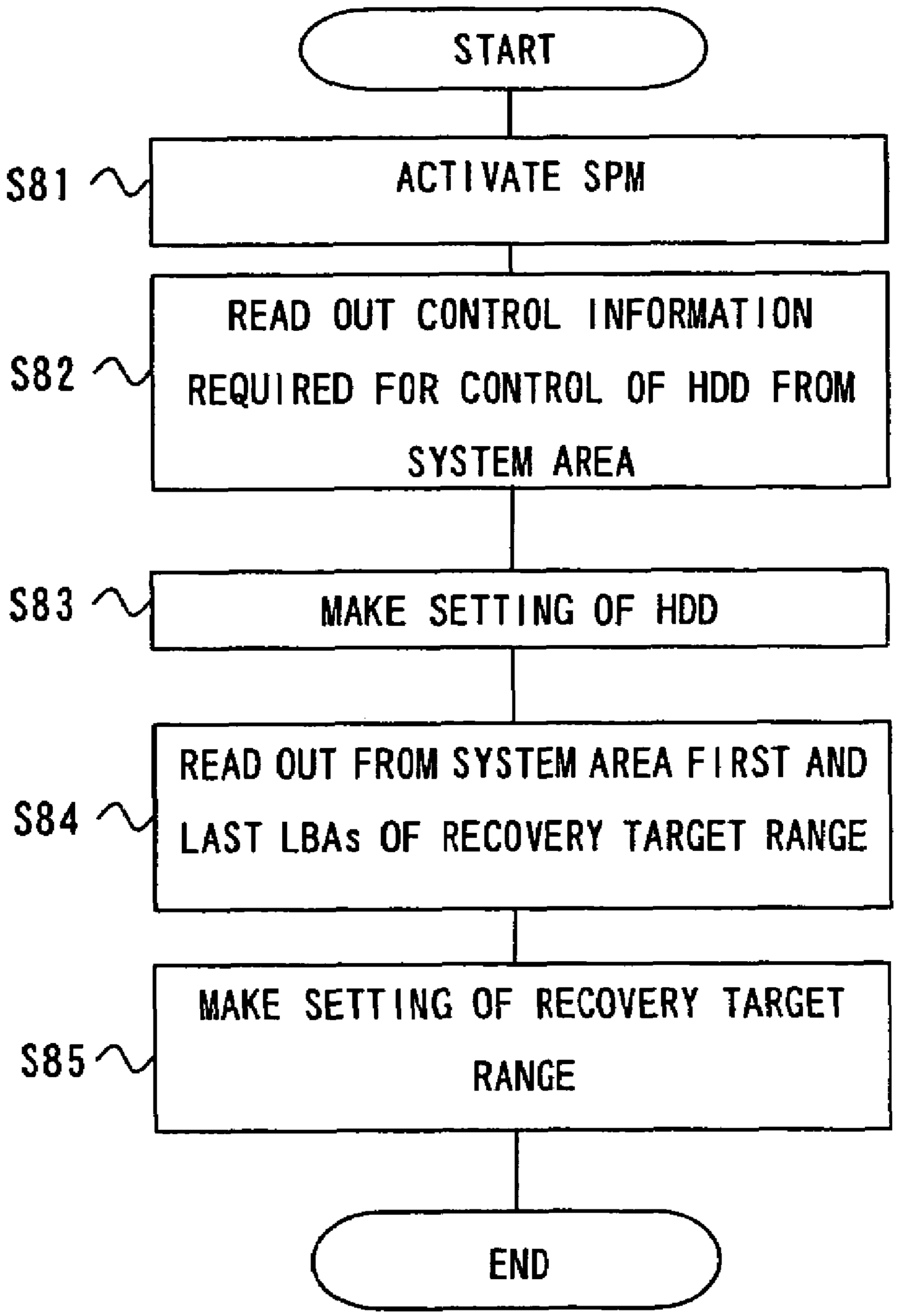

START
S81
ACTIVATE SPM
S82
READ OUT CONTROL INFORMATION REQUIRED FOR CONTROL OF HDD FROM SYSTEM AREA
S83
MAKE SETTING OF HDD
S84
READ OUT FROM SYSTEM AREA FIRST AND LAST LBAs OF RECOVERY TARGET RANGE
S85
MAKE SETTING OF RECOVERY TARGET RANGE
END

FIG. 14

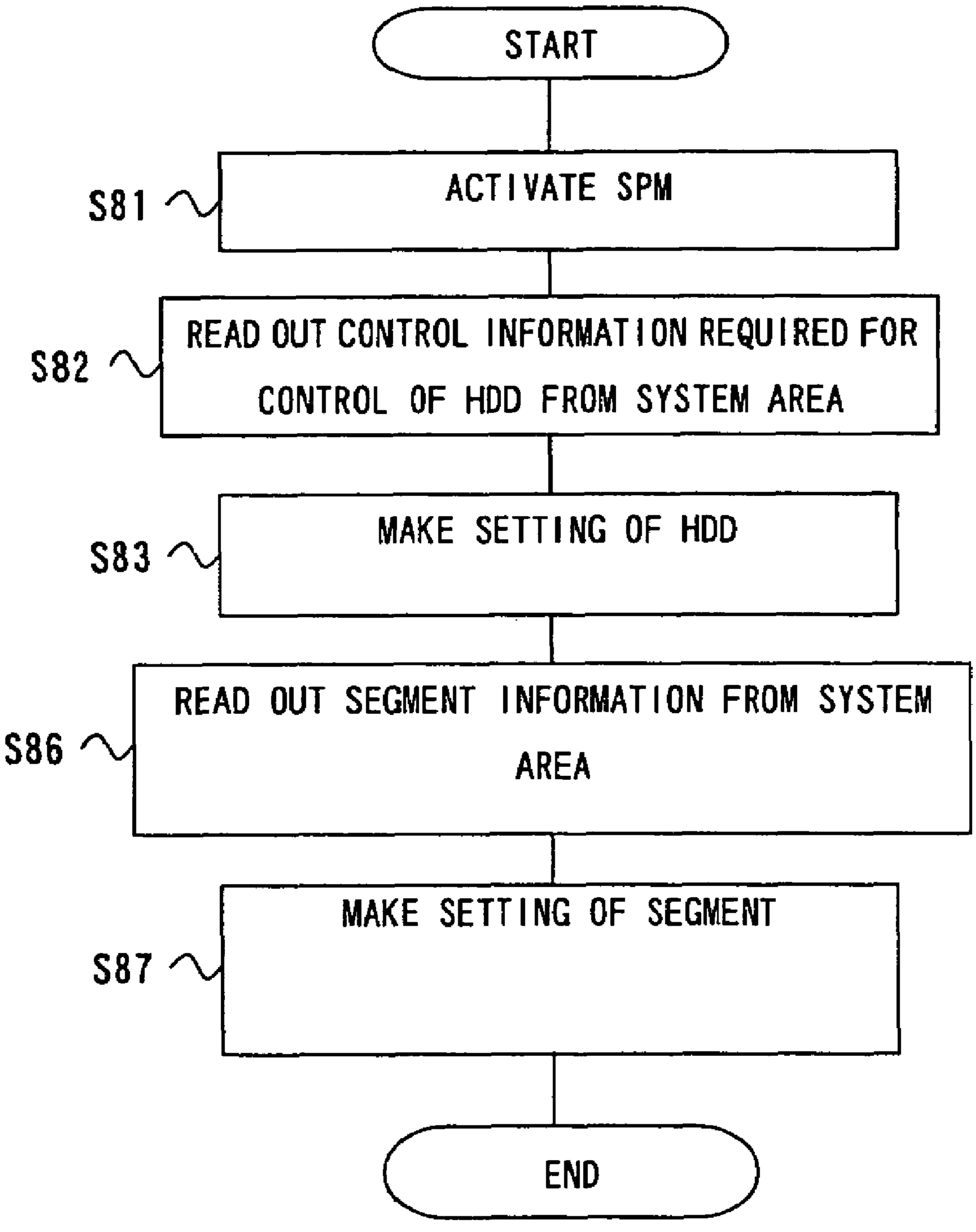

START
S81
ACTIVATE SPM
S82
READ OUT CONTROL INFORMATION REQUIRED FOR CONTROL OF HDD FROM SYSTEM AREA
S83
MAKE SETTING OF HDD
S86
READ OUT SEGMENT INFORMATION FROM SYSTEM AREA
S87
MAKE SETTING OF SEGMENT
END

FIG. 15

| SEGMENT START LBA | PRESENCE/ABSENCE OF REDUNDANT SECTOR DATA | SEGMENT SIZE |
|---|---|---|
| 0x00000000 | PRESENT | 0x0800 |
| 0x00010000 | PRESENT | 0x10000 |
| 0x00200000 | ABSENT | |
| 0xffffffff | ABSENT | |

FIG. 16

| | |
|---|---|
| REDUNDANT0 | REDUNDANT 0 VALID FLAG |
| REDUNDANT1 | REDUNDANT 1 VALID FLAG |
| REDUNDANT2 | REDUNDANT 2 VALID FLAG |
| REDUNDANT3 | REDUNDANT 3 VALID FLAG |
| ... | ... |
| ... | ... |
| REDUNDANT n | REDUNDANT n VALID FLAG |

FIG. 17

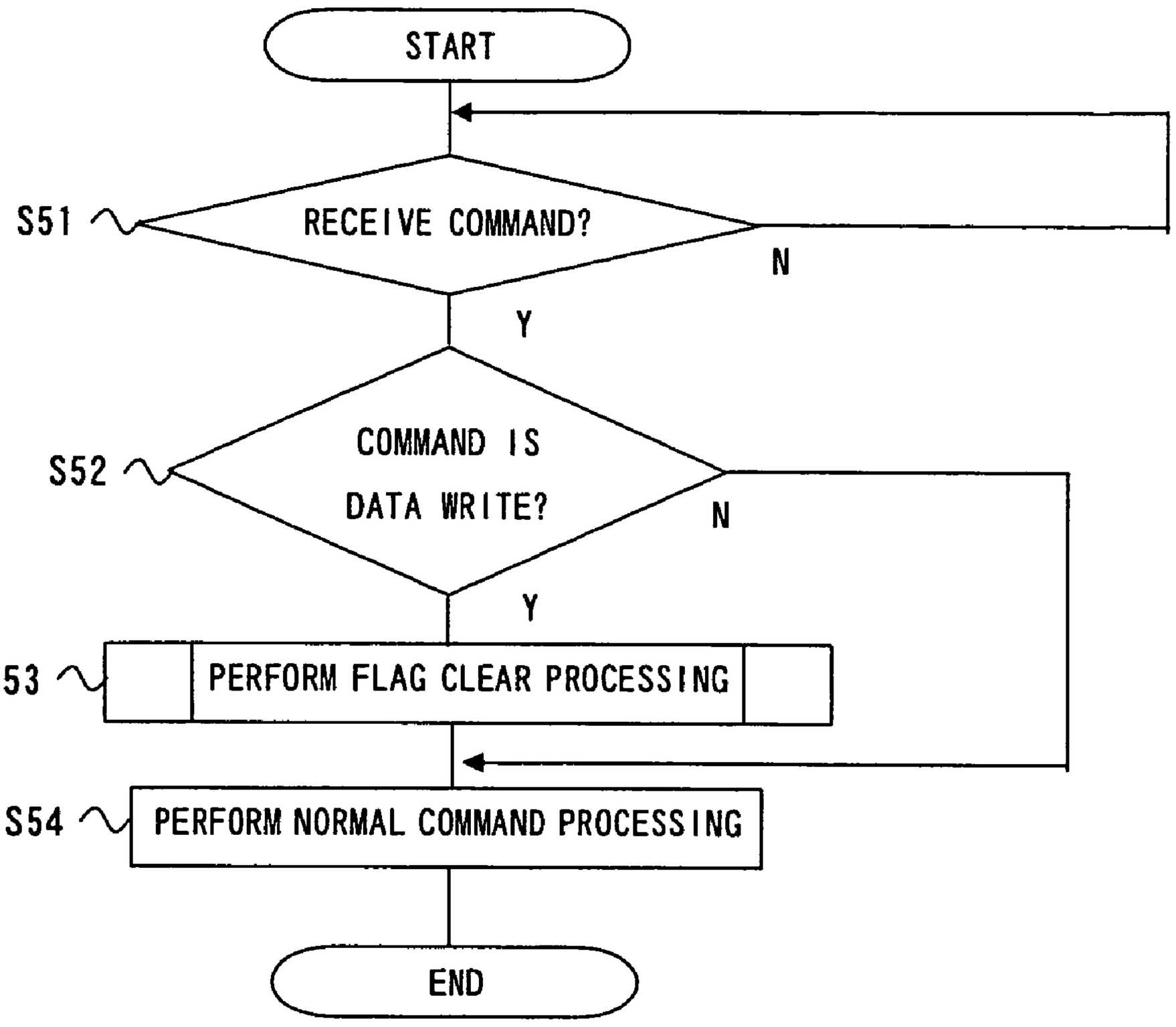

START
S51
RECEIVE COMMAND?
N
Y
S52
COMMAND IS DATA WRITE?
N
Y
S53
PERFORM FLAG CLEAR PROCESSING
S54
PERFORM NORMAL COMMAND PROCESSING
END

FIG. 18

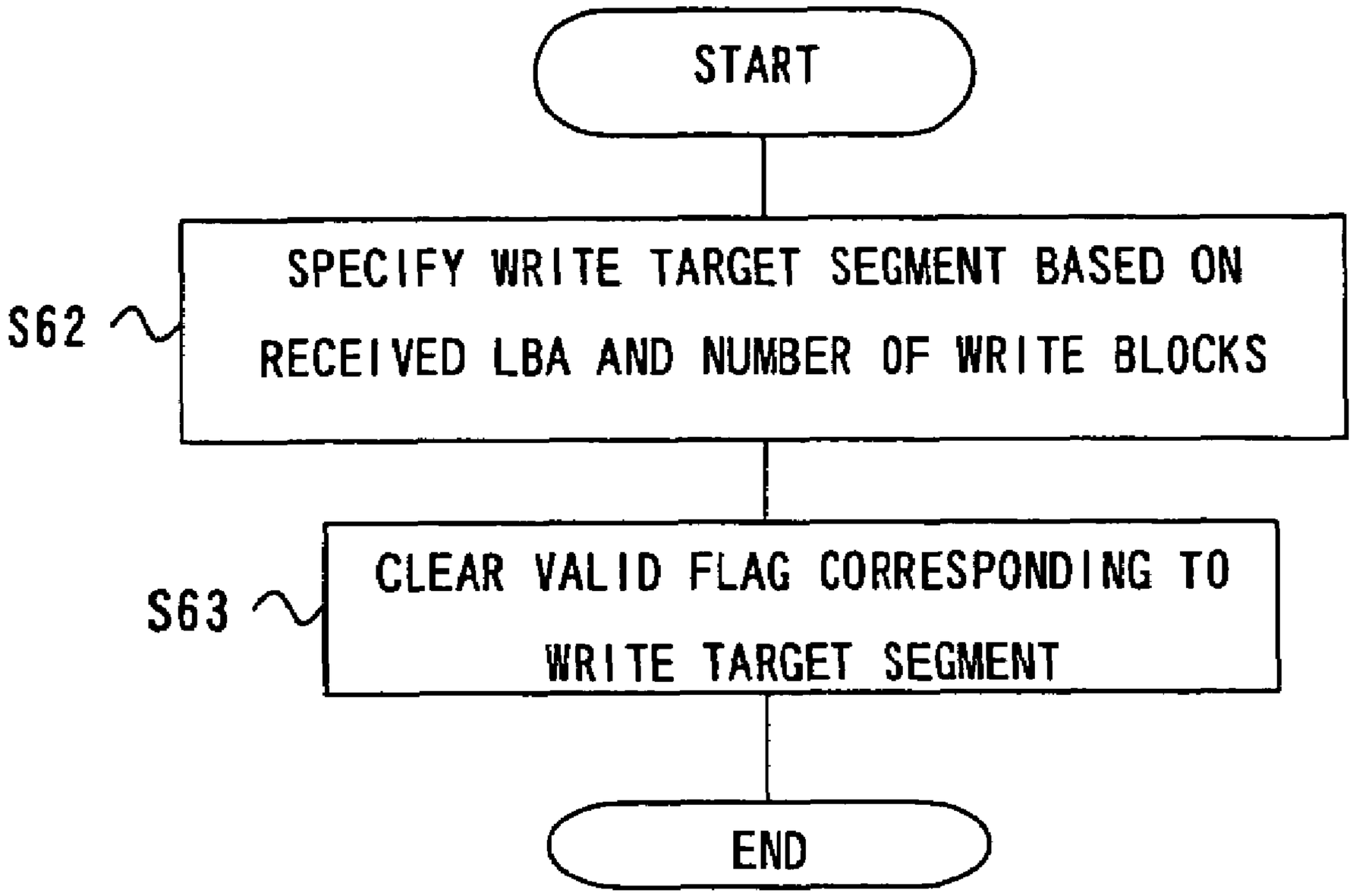

START
S62
SPECIFY WRITE TARGET SEGMENT BASED ON RECEIVED LBA AND NUMBER OF WRITE BLOCKS
S63
CLEAR VALID FLAG CORRESPONDING TO WRITE TARGET SEGMENT
END

FIG. 19

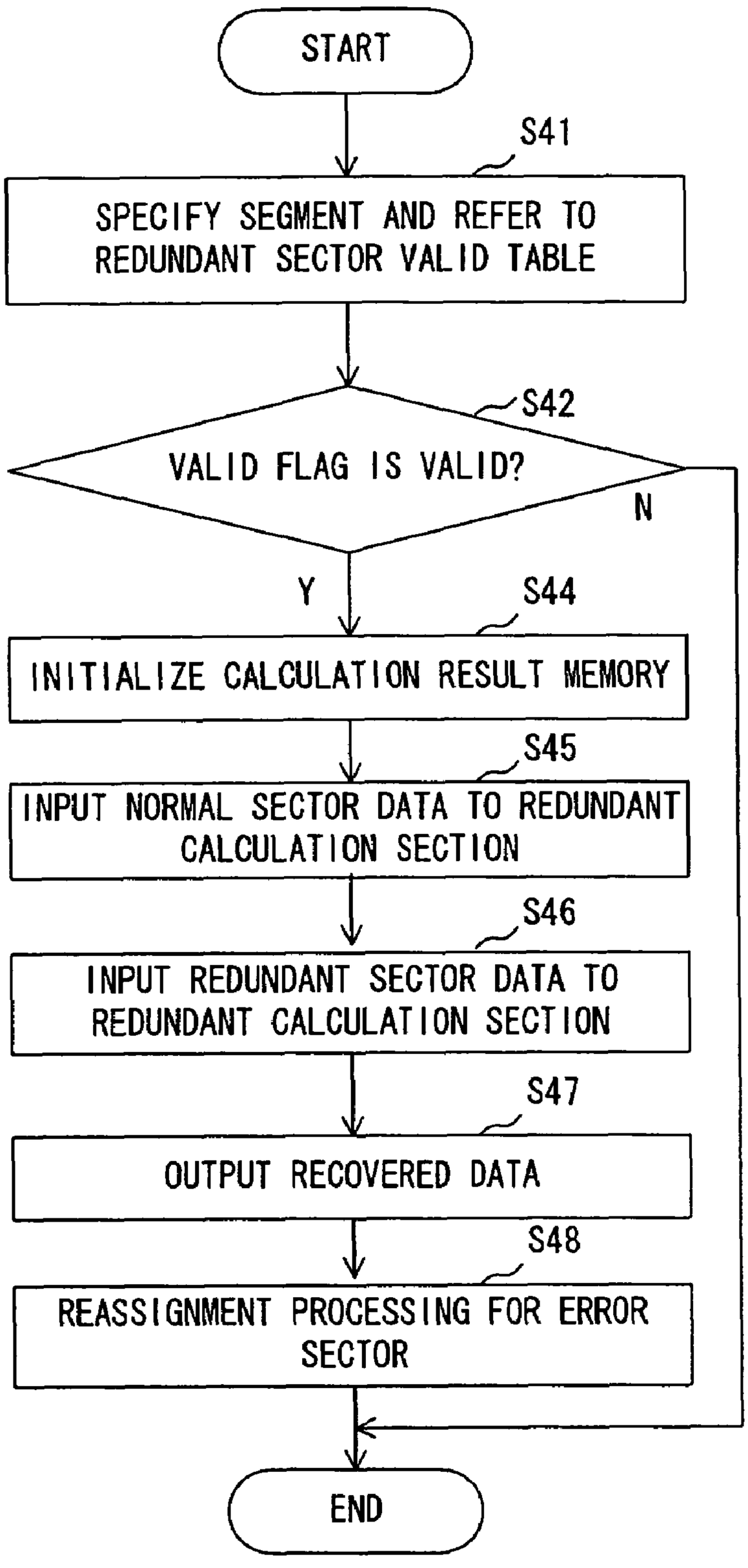
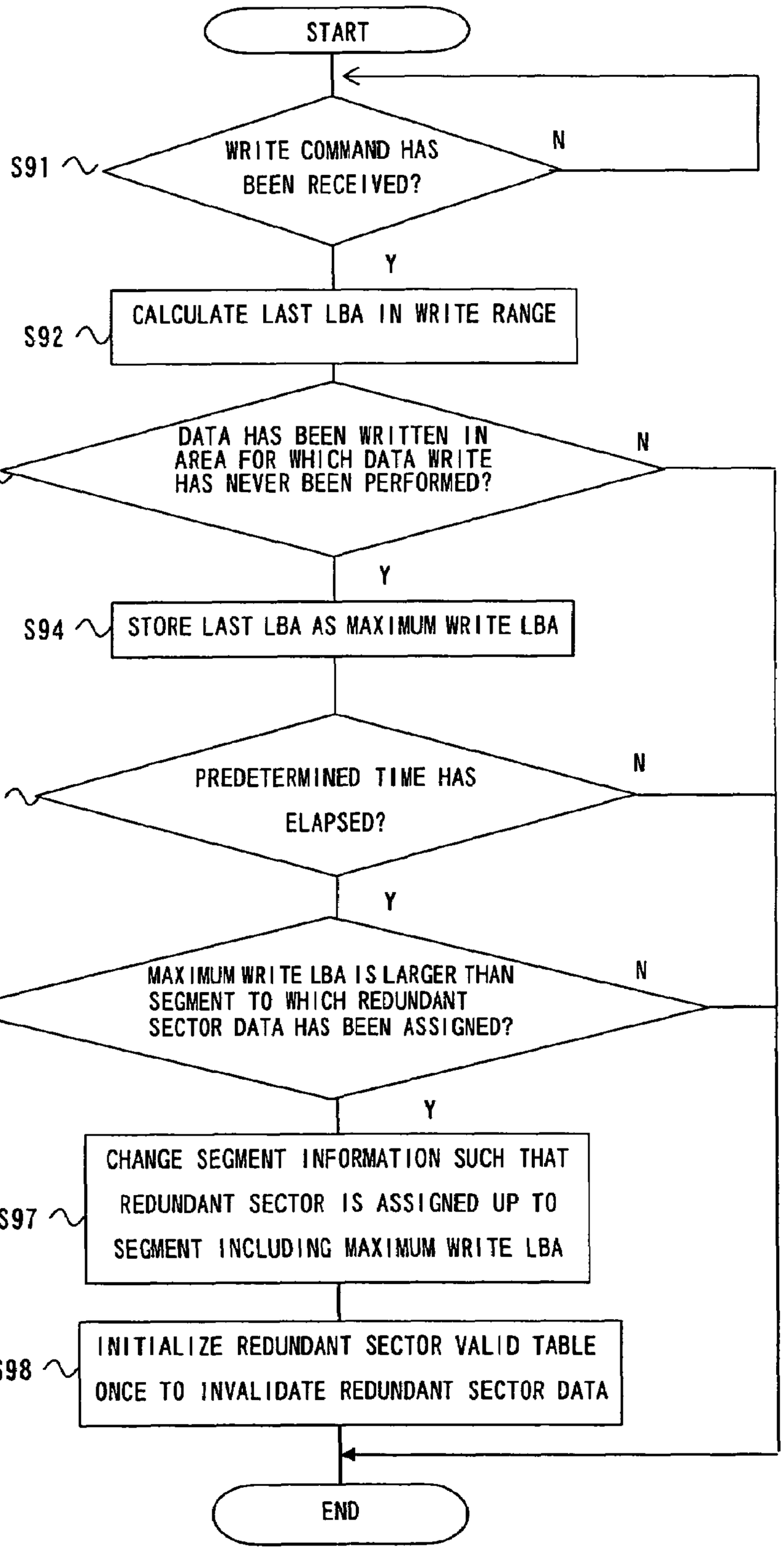

START
S41
SPECIFY SEGMENT AND REFER TO REDUNDANT SECTOR VALID TABLE
S42
VALID FLAG IS VALID?
N
Y
S44
INITIALIZE CALCULATION RESULT MEMORY
S45
INPUT NORMAL SECTOR DATA TO REDUNDANT CALCULATION SECTION
S46
INPUT REDUNDANT SECTOR DATA TO REDUNDANT CALCULATION SECTION
S47
OUTPUT RECOVERED DATA
S48
REASSIGNMENT PROCESSING FOR ERROR SECTOR
END

FIG. 20


START
S91
WRITE COMMAND HAS BEEN RECEIVED?
N
Y
S92
CALCULATE LAST LBA IN WRITE RANGE
S93
DATA HAS BEEN WRITTEN IN AREA FOR WHICH DATA WRITE HAS NEVER BEEN PERFORMED?
N
Y
S94
STORE LAST LBA AS MAXIMUM WRITE LBA
S95
PREDETERMINED TIME HAS ELAPSED?
N
Y
S96
MAXIMUM WRITE LBA IS LARGER THAN SEGMENT TO WHICH REDUNDANT SECTOR DATA HAS BEEN ASSIGNED?
N
Y
S97
CHANGE SEGMENT INFORMATION SUCH THAT REDUNDANT SECTOR IS ASSIGNED UP TO SEGMENT INCLUDING MAXIMUM WRITE LBA
S98
INITIALIZE REDUNDANT SECTOR VALID TABLE ONCE TO INVALIDATE REDUNDANT SECTOR DATA
END

# STORED DATA PROCESSING APPARATUS, STORAGE APPARATUS, AND STORED DATA PROCESSING PROGRAM

## FIELD OF THE INVENTION

The present invention relates to a storage apparatus that performs recovery operations associated with a read error and a stored data processing apparatus and stored data processing program used for the storage apparatus.

## DESCRIPTION OF THE RELATED ART

The recording capacity of an HDD (Hard Disk Drive) has been increasing, and hard disks of 200 G bytes or more are now available. Along with the increase in recording capacity, the degree of damage becomes severe in the case where a read error occurs in one given sector. In particular, in the case where a read error occurs in one given sector corresponding to a location where directory information concerning to data management in an HDD has been recorded, enormous numbers of data may be damaged. The seriousness of the damage increases as the recording amount increases.

As a means for avoiding troubles specific to such an HDD, a system in which one HDD is configured as part of an array has been available. In this system, if there exists any drive that has got a read error, another HDD provided for increasing redundancy is used to recover the drive data. In order to enhance the resistance of a single HDD against such an error, a method that adds an ECC (Error Correcting Code) to the information in the sector has been adopted. Each sector on a disk medium is composed of SB (syncbyte), DATA, and ECC.

Assume that a read error occurs in a given sector. At this time, a data buffer in an HDD retains data received from a sector at which readout operation is started to a sector which is one sector before a sector where the read error has occurred. If the data readout operation from the sector at which the read error has occurred does not succeed even when retry is attempted, the HDD cannot load effective data into the data buffer and therefore cannot send the effective data to a higher-level device. When the data recovery fails eventually, the HDD notifies the higher-level device of a read error and ends the readout operation.

As a prior art relating to the present invention, there is known a file control apparatus (refer to e.g., Patent Document 1: Jpn. Pat. Appln. Laid-Open Publication No. 5-35416) that prepares a buffer corresponding to the size of data of each the track and performs data check processing in units of tracks. Further, there is known a data recording method (refer to e.g., Patent Document 2: Jpn. Pat. Appln. Laid-Open Publication No. 8-124318) that adds a parity forming LCD (Long Distance Code) to each sector, adds parity sector data to each predetermined number of sectors, calculates parity sector data for each predetermined number of sectors, and uses the parity sector data to make correction when correction using the parity forming the LDC is disabled at reproduction time.

Although the error correction capability itself using the ECC gradually increases, a new data correction method has not been adopted and conventional techniques have continued to be used in recent years. However, in a state where the recording capacity of a single HDD has been increasing, it can be said that the risk of data loss relatively increases with the conventional data error correction capability using the ECC.

Further, as described above, the mechanism in which, in the case where error correction cannot be made using the ECC, the conventional HDD notifies the higher-level device of a read error and ends the readout operation remains basically unchanged since the birth of a disk drive.



Further, in the technique like Patent Document 1, data corresponding to one track needs to be read out every time one writing processing is performed, thus doubling command processing time to deteriorate the performance. As a result, the use of this technique is not realistic. Further, in the technique like Patent Document 2, redundant data needs to be prepared when write data is received from the higher-level device to thereby deteriorate the performance, making it difficult to use this technique practically.

## SUMMARY OF THE INVENTION

The present invention has been made to solve the above problems, and an object thereof is to provide a storage apparatus capable of enhancing error recovery capability and a stored data processing apparatus and stored data processing program used for the storage apparatus.

To solve the above problem, according to a first aspect of the present invention, there is provided a stored data processing apparatus for use in a storage apparatus that performs at least write processing for a storage medium on which data is allocated in units of a first block having a predetermined data length, the first block is allocated in units of a second block constituted by a plurality of the first blocks, and a third block having the same data length as the first block is allocated in correspondence with the second block, comprising: a management information storage section that stores, for each second block, management information including information indicating whether the third block corresponding to the second block is valid or invalid; a calculation section that performs calculation for each bit position using data of all the first blocks in an update target in the case where the management information stored in the management information storage section indicates that the third block corresponding to the second block specified as the update area is invalid and outputs a result of the calculation as calculation data; and a write section that writes the calculation data output from the calculation section in the third block associated with the update target and validates the third block in the management information stored in the management information storage section.

In the stored data processing apparatus according to the present invention, the calculation section initializes the calculation data, sequentially sets data of the respective first blocks in the update target as input data of the calculation, performs calculation for each bit position using the input data and calculation data, stores a result of the calculation as new calculation data, and repeats the calculation and storage for all the input data to output the calculation data.

The stored data processing apparatus according to the present invention further comprises a read section that reads out data from the first block that has been specified as a read target, sets the first block from which data has not normally been read out as an error block in the case where the data read has failed, reads out normal data which is data of all the first blocks other than the error block in the second block in the case where the management information stored in the management information storage section indicates that the third block corresponding to the second block including the read target, and reads out redundant data which is data of the third block corresponding to the second block, wherein the calculation section further performs calculation for each bit position using the normal data and redundant data and outputs a result of the calculation as data of the error block.

In the stored data processing apparatus according to the present invention, the calculation section initializes the calculation data, sequentially sets data of the respective normal data and redundant data as input data of the calculation, performs calculation for each bit position using the input data and calculation data, stores a result of the calculation as new calculation data, and repeats the calculation and storage for all the input data to output the calculation data.

In the stored data processing apparatus according to the present invention, when receiving a data write instruction from an external device, the write section validates the third block corresponding to the second block including the data write target specified by the data write instruction in the management information stored in the management information storage section.

In the stored data processing apparatus according to the present invention, the calculation section performs the calculation for the second block corresponding to the third block which is invalidated in the management information stored in the management information storage section while it does not perform processing based on a data read instruction or data write instruction from an external device.

In the stored data processing apparatus according to the present invention, the management information storage section is a non-volatile memory.

In the stored data processing apparatus according to the present invention, the management information storage section writes management information onto the storage medium at a timing of receiving, from an external device, an instruction of writing write cache data onto the storage medium or instruction of retreating a head.

In the stored data processing apparatus according to the present invention, the management information includes specification of a plurality of first blocks included in the second block, which is made for each second block.

In the stored data processing apparatus according to the present invention, the management information includes specification of a plurality of first blocks included in the second block, which is made for each second block, and the write section allocates the second block on the storage medium based on the management information.

In the stored data processing apparatus according to the present invention, the management information storage section determines the number of first blocks included in the second block based on the access frequency from an external device to each second block so as to set the determined number in the management information.

In the stored data processing apparatus according to the present invention, the calculation performed by the calculation section is a calculation in which the same calculation result can be obtained even if the order of the calculation is changed.

In the stored data processing apparatus according to the present invention, the calculation performed by the calculation section is exclusive OR.

In the stored data processing apparatus according to the present invention, the first block is at least one sector, the third block has the same data length as the first block, and the calculation data has the same data length as the first block.

According to a second aspect of the present invention, there is provided a stored data processing program allowing a computer to execute stored data processing in a storage apparatus, comprising: a management information storage step that stores, for each second block, management information including information indicating whether the third block corresponding to the second block is valid or invalid, the information being related to a storage medium on which data is allocated in units of a first block having a predetermined data length, the first block is allocated in units of a second block constituted by a plurality of the first blocks, and a third block having the same data length as the first block is allocated in correspondence with the second block; a calculation step that performs calculation for each bit position using data of all the first blocks in an update target in the case where the management information stored in the management information storage step indicates that the third block corresponding to the second block specified as the update area is invalid and outputs a result of the calculation as calculation data; and a write step that writes the calculation data output from the calculation step in the third block associated with the update target and validates the third block in the management information stored in the management information storage step.

According to a third aspect of the present invention, there is provided a storage apparatus that performs at least write processing for a storage medium on which data is allocated in units of a first block having a predetermined data length, the first block is allocated in units of a second block constituted by a plurality of the first blocks, and a third block having the same data length as the first block is allocated in correspondence with the second block, comprising: a management information storage section that stores, for each second block, management information including information indicating whether the third block corresponding to the second block is valid or invalid; a calculation section that performs calculation for each bit position using data of all the first blocks in an update target in the case where the management information stored in the management information storage section indicates that the third block corresponding to the second block specified as the update area is invalid and outputs a result of the calculation as calculation data; and a write section that writes the calculation data output from the calculation section in the third block associated with the update target and validates the third block in the management information stored in the management information storage section.

According to the present invention, there can be provided a storage apparatus capable of enhancing error recovery capability and increasing data reliability in accordance with advancement in high-density recording, and a stored data processing apparatus and stored data processing program for use in the storage apparatus.

## BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** is a block diagram showing an example of a configuration of an HDD according to a first embodiment;

FIG. **2** is a block diagram showing an example of a configuration of a redundant calculation section according to the first embodiment;

FIG. **3** is an allocation diagram showing an example of sector allocation in the user area on the disk medium according to the first embodiment;

FIG. **4** is a flowchart showing an example of operation of redundant sector data update processing according to the first embodiment;

FIG. **5** is a flowchart showing an example of operation of redundant sector data generation processing according to the first embodiment;

FIG. **6** is a flowchart showing an example of operation of single redundant sector data write processing according to the first embodiment;

FIG. **7** is a flowchart showing an example of operation of the error sector data recovery processing according to the first embodiment;

FIG. **8** is an allocation diagram showing an example of sector allocation in the user area on the disk medium according to a second embodiment;

FIG. **9** is a flowchart showing an example of operation of the redundant sector data update processing according to the second embodiment;

FIG. **10** is an allocation diagram showing an example of sector allocation in the user area on the disk medium according to a third embodiment;

FIG. **11** is a block diagram showing an example of a configuration of an HDD according to a fifth embodiment;

FIG. **12** is a flowchart showing an example of the single redundant sector data write processing according to the fifth embodiment;

FIG. **13** is a flowchart showing an example of operation of recovery target range identification processing according to a sixth embodiment;

FIG. **14** is a flowchart showing an example of operation of segment size identification processing according to a seventh embodiment;

FIG. **15** is a table showing an example of segment information according to the seventh embodiment.

FIG. **16** is a table showing an example of the data structure of a redundant sector data table according to an eighth embodiment;

FIG. **17** is a flowchart showing an example of operation of command reception processing according to the eighth embodiment;

FIG. **18** is a flowchart showing an example of operation of flag clear processing according to the eighth embodiment;

FIG. **19** is a flowchart showing an example of operation of the error sector data recovery processing according to the eighth embodiment; and

FIG. **20** is a flowchart showing an example of operation of segment information change processing according to a ninth embodiment.

## DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

Embodiments of the present invention will be described below with reference to the accompanying drawings.

### First Embodiment

A first embodiment shows an example in which a stored data processing apparatus and a storage apparatus according to the present invention have been applied to an HDD (Hard Disk Drive).

A configuration of the HDD according to the present embodiment will first be described.

FIG. **1** is a block diagram showing an example of a configuration of the HDD according to the present embodiment.

A conventional HDD will next be described. This HDD includes a host IF (Interface) controller **2** that controls a host IF for connection to an external device, a data buffer controller **3**, a data buffer **4**, a format controller **5**, a read channel **6**, a head IC (Integrated Circuit) **7**, an MPU (Micro Processing Unit) for control of the HDD, a memory **9** that stores control data and a control program, a non-volatile memory (FROM) **10** that stores the control program, a servo controller **11** that controls the operation of a spindle motor (SPM) or voice coil motor (VCM), a VCM **12** that actually moves a head actuator, a SPM **13** that rotates a disk, a head **14** for Read/Write, a disk medium **15**, an internal common bus **16**, and a redundant calculation section **21**.

The HDD is connected to a higher-level device (external device) such as a PC (Personal Computer) via the host IF connected to the host IF controller **2**. The host IF controller **2**, data buffer controller **3**, format controller **5**, read channel **6**, head IC **7**, MPU **8**, memory **9**, non-volatile memory **10**, and servo controller **11** are connected with one another via the common bus **16**.

In data reading operation from the HDD to the higher-level device, data is read out form the disk medium **15** according to a data read instruction from the higher-level device and is sent to the read channel **6** through the head **14** and head IC **7**. The data is further sent to the format controller **5**, where ECC calculation is performed. Only the data in which no error has detected is temporarily stored in the data buffer **4** through the data buffer controller **3**. Then, the data is sent to the higher-level device through the data buffer controller **3** and host IF controller **2**.

In data writing operation from the higher-level device to the HDD, data sent from the higher-level device as a write command is sent via the host IF controller **2** and data buffer controller **3** to the data buffer **4**, where the data is temporarily loaded thereinto. Further, at a timing appropriate for execution of the data writing, the data is sent via the data buffer controller **3**, format controller **5**, read channel **6**, and head IC **7** to the disk medium **15**, where the data is written thereon.

FIG. **2** is a block diagram showing an example of a configuration of the redundant calculation section according to the present embodiment. The redundant calculation section **21** includes a calculation result memory **31** and an XOR (Exclusive OR) calculation section **32**. The calculation result memory **31** has a capacity corresponding to a single sector size and retains calculation result data (calculation data) corresponding to one sector. Input data corresponding to one sector from an external device and calculation result data corresponding to one sector retained by the calculation result memory **31** are XORed for each bit position by the XOR calculation section **32**. The result of the XOR calculation is retained by the calculation result memory **31** as a new calculation result data. The MPU **8** can read out the calculation result data retained by the calculation result memory **31** via the common bus **16** to set the calculation result as redundant sector data.

Sector allocation in the user area on the disk medium **15** will next be described.

FIG. **3** is an allocation diagram showing an example of sector allocation in the user area on the disk medium according to the present embodiment. This diagram shows the allocation of normal sectors into which normal user data are written and redundant sectors into which redundant sector data are written. One element in this diagram represents one sector. Further, in this diagram, a segment is represented by a segment number, a normal sector is represented by an LBA (Logical Block Address), and a redundant sector is represented by a corresponding segment number. Here, normal sectors having m consecutive LBAs are defined as one segment, and the one segment is represented by one row. The entire user area is divided into n segments. One redundant sector into which redundant sector data generated from the user data of one segment are written is inserted from each segment. That is, one redundant sector data (j) is created for the user data having j-th segment LBA (ji) to LBA (j+1) i−1) for insertion (i=0, 1, . . . m−1) (j=0, 1, . . . n−1).

Operation of the HDD according to the present embodiment will next be described.

Redundant sector data update processing that creates redundant sector data from the user data on the disk medium **15** and performs writing of the created redundant sector data will first be described.

The redundant sector data update processing is executed when an update instruction is received from the higher-level device or when the time period during which no access has been made from the higher-level device (spare time of the MPU **8**) exceeds a threshold value. FIG. **4** is a flowchart showing an example of operation of the redundant sector data update processing according to the present embodiment. The MPU **8** sets the entire user area as an update range (S**12**). The MPU **8** then performs redundant sector data generation processing for each segment in the set update range (S**13**) and single redundant sector data write processing to write generated one redundant data onto the disk medium **15** (S**14**). Subsequently, the MPU **8** determines whether the redundant sector data of all segments in the update range have been updated. If all the redundant sector data have not been updated (N in S**17**), the MPU **8** returns to step S**13**, while if all the redundant sector data have been updated (Y in S**17**), the MPU **8** ends this flow.

Redundant sector data generation processing of step S**13** will next be described.

The redundant sector data generation processing is processing that reads out user data of a target segment and generates redundant sector data from the read out user data by the redundant calculation section **21**. FIG. **5** is a flowchart showing an example of operation of the redundant sector data generation processing according to the present embodiment. The MPU **8** sets the first and last LBAs of a target segment, number of sectors to be processed, and sector allocation are set in the format controller **5** (S**21**). The sector allocation represents the allocation of the normal sectors used for generation of the redundant sector data and specifies the order of the normal sector to be read out. In the present embodiment, the normal sectors of a target segment are allocated consecutively as described above, and the MPU **8** specifies the sector allocation so as to read out the consecutively allocated normal sectors. Then, the MPU **8** initializes (zero-clears) the calculation result memory **31** of the redundant calculation section **21** (S**22**). Then, the MPU **8** activates the format controller **5** to allow the format controller **5** to read out target user data and to input the read out user data to the redundant calculation section **21** (S**23**). After that, the MPU **8** determines whether the operation has been completed. If the operation has not been completed (N in S**24**), the MPU **8** returns to step S**24**, while if the operation has been completed (Y in S**24**), the MPU **8** ends this flow.

Operation of the redundant calculation section **21** during the redundant sector data generation processing will next be described. The data corresponding to one sector retained in the calculation result memory **31** are all initialized to zero, so that the input data which is the first user data are retained without change as the calculation result data in the calculation result memory **31**. Subsequent user data and calculation result data are XORed for each bit position and overwritten as the calculation result data in the calculation result memory **31**. At the time point at which the redundant sector data generation processing has been completed, the calculation result data retained in the calculation result memory **31** become the redundant sector data.

The last segment which is a segment including the last LBA may have a smaller number of normal sectors than that in the other segments in some cases. However, if only all the user data in the segment are input as input data in the redundant sector data generation processing, the redundant sector data can be obtained, irrespective of the number of the normal sectors.

The single redundant sector data write processing will next be described.

FIG. **6** is a flowchart showing an example of operation of the single redundant sector data write processing according to the present embodiment. The MPU **8** copies the redundant sector data retained in the calculation result memory **31** to the data buffer **4** (S**31**). Then the MPU **8** writes the redundant sector data in the data buffer **4** onto the disk medium **15** (S**32**) in the same manner as the processing of writing normal user data in the data buffer **4** onto the disk medium **15** and ends this flow.

In the same manner as writing conventional user data, by using a path for writing user data from the data buffer **4** onto the disk medium **15** as a path for writing the redundant sector data, it is possible to suppress an increase in the hardware scale in the case where the present invention is applied to a conventional HDD. Further, also in update processing of the second redundant sector data, by coping all the redundant sector data temporarily to the data buffer **4**, all the redundant sector data can be written onto the disk medium **15** in a consecutive manner.

As described above, it is preferable that the update of the redundant sector data be performed in a timing other than a reception timing of a command (write command or read command) from the higher-level device. A case where an unreadable error sector occurs in an HDD is so rare that it is not necessary to update the redundant sector data at the reception timing of the command from the higher-level device or a timing immediately after the command reception timing. It is because that increased frequency of the redundant sector data update processing increases processing overhead, thereby deteriorating the performance of the HDD.

User data readout processing and disk medium scan processing will next be described.

In the case where the MPU **8** has received a read command from the higher-level device and has normally read out data of all target sectors, the processing concerning the redundant sector data is not required and, in this case, it is only necessary to send the target user data to the higher-level device. On the other hand, if there found any error sector which cannot be recovered in the user data readout processing, error sector data recovery processing that recovers error sector data which is data that have been written in an error sector is performed.

In the case where the MPU **8** receives a disk medium scan command from the higher-level device, the MPU **8** performs disk medium scan processing by reading out data of all sectors in the user area on the disk medium **15** in the same manner as the user data readout processing to detect an error sector. If there found any error sector in the disk medium scan processing, the error sector data recovery processing is performed as in the case of the user data readout processing.

The MPU **8** performs the disk medium scan processing by reading out data of all sectors in the user area on the disk medium **15** in the same manner as the user data readout processing at a predetermined time period. If there found any error sector in the disk medium scan processing, the error sector data recovery processing is performed as in the case of the user data readout processing.

The error sector data recovery process will next be described.

FIG. **7** is a flowchart showing an example of operation of the error sector data recovery processing according to the present embodiment. The MPU **8** initializes the calculation result memory **31** (S**44**), reads out from the disk medium **15**

normal sectors in the segment to which the found error sector belongs, and inputs readable user data (user data other than the error sector data) to the redundant calculation section **21** (S**45**). Then, the MPU **8** reads out from the disk medium **15** the redundant sector data in the segment to which the found error sector belongs and input the readout redundant sector data to the redundant calculation section **21** (S**46**). At this time, through the operation of the redundant calculation section **21**, the calculation result data retained in the calculation result memory **31** becomes recovered data in which the data of the error sector has been recovered. The MPU **8** then reads out the recovered data from the calculation result memory **31** and sends the read out recovered data to the higher-level device (S**47**). After that, the MPU **8** performs reassignment processing to assign an empty sector to the error sector (S**48**) and ends this flow.

As described above, by performing XOR calculation for the user data in the target segment other than the error sector data and redundant sector data, error sector data can be recovered. Further, even if the order of the sectors to be input as input data is changed, the same calculation result data can be obtained. Therefore, no matter what sector order the user data of the normal sectors in a given segment are input, the XOR calculation section **32** can obtain redundant sector data as the final calculation result data and, no matter what sector order the user data other than the error sector data and redundant sector data in the segment in which the error sector exists are input, the XOR calculation section **32** can obtain error sector data as the final calculation result data. Further, even with respect to a segment, like the last segment, having a smaller number of normal sectors than that in the other segments, the redundant sector data generation processing and error sector data recovery processing can be performed as in the case of the other segments by using the user data of all normal sectors.

Although the XOR calculation section **32** is used for performing the redundant sector data generation processing and error sector data recovery processing in the present embodiment, other calculations may be employed as far as they are calculation methods in which the same calculation result can be obtained even if the order of a plurality of sectors to be input as input data is changed. Further, although the input data and calculation result data are treated in units of sector, they may be treated in units of other data lengths such as a byte and word.

According to the present embodiment, even if a read error has occurred, error sector data can be recovered. Thus, reliability of a disk drive is remarkably increased. Further, by performing update of the redundant data sector according to an instruction from the higher-level device and spare time of the MPU **8**, an increase in response time for the higher-level device can be suppressed.

## Second Embodiment

In a second embodiment of the present invention, an HDD capable of collectively writing a plurality of redundant sector data will be described.

The HDD according to the present embodiment has the same configuration as that of the HDD according to the first embodiment.

The sector allocation in the user area on the disk medium **15** will next be described.

FIG. **8** is an allocation diagram showing an example of sector allocation in the user area on the disk medium according to the present embodiment. In this case, all redundant sectors are consecutively allocated, following after all normal sectors shown in the first embodiment.

Operation of the HDD according to the present embodiment will next be described.

A difference in the HDD operation between the first and second embodiments lies in the redundant sector data update processing. FIG. **9** is a flowchart showing an example of operation of the redundant sector data update processing according to the present embodiment. In FIG. **9**, the same reference numerals denote the same or corresponding parts as in FIG. **4**, and the descriptions thereof are omitted here. After step S**13**, the MPU **8** temporarily stores in the data buffer **4** the redundant sector data retained in the calculation result memory **31** such that respective generated redundant sector data are not overlapped with one another (S**15**). Then, the MPU **8** determines whether all the redundant sector data in the update range have been stored in the data buffer **4** (S**18**). If all the redundant sector data in the update range have not yet been stored in the data buffer **4** (N in S**18**), the MPU **8** returns to step S**13**, while if all the redundant sector data in the update range have already been stored in the data buffer **4** (Y in S**18**), the MPU **8** collectively writes all the redundant sector data in the data buffer **4** onto the disk medium **15** (S**19**) and ends this flow.

According to the present embodiment, a plurality of redundant sector data can be written in a consecutive manner onto the disk medium, thereby reducing the time needed to write the redundant sector data.

## Third Embodiment

In a third embodiment of the present invention, an HDD capable of recovering data of the error sector even in the case where two consecutive error sectors exist in one segment will be described.

The HDD according to the present embodiment has the same configuration as that according to the first embodiment.

The sector allocation in the user area on the disk medium **15** will next be described.

FIG. **10** is an allocation diagram showing an example of sector allocation in the user area on the disk medium according to the present embodiment. This diagram shows the allocation of the normal sectors and redundant sectors. Further, in this diagram, a segment is represented by a segment number, a normal sector is represented by an LBA assigned thereto, and a redundant sector is represented by a corresponding segment number. Here, user data having 2m consecutive LBAs are alternately allocated between two segments, and one redundant sector is inserted for each segment.

For example, m normal sectors having even-numbered LBAs (LBA (**0**), LBA (**2**), . . . LBA (m−2)) out of normal sectors having m consecutive LBAs (LBA (**0**) to LBA (m−1)) and odd-numbered LBAs (LBA (m+1), LBA (m+3), . . . LBA (2m−1)) out of the following normal sectors having m consecutive LBAs (LBA (m) to LBA (2m−1)) are alternately allocated. A segment represented as a first raw is defined as segment **0**, and a redundant sector **0** which is obtained from the m normal sector in this segment **0** is inserted after the end of the segment **0**. Similarly, m normal sectors having even-numbered LBAs (LBA (m), LBA (m+2), . . . LBA (2m−2)) out of normal sectors having m consecutive LBAs (LBA (m) to LBA (2m−1)) and odd-numbered LBAs (LBA (**1**), LBA (**3**), . . . LBA (m−1)) out of normal sectors having m consecutive LBAs (LBA (**0**) to LBA (m−1)) are alternately allocated. A segment represented as a second raw is defined as segment

1, and a redundant sector 1 which is obtained from the m normal sector in this segment 1 is inserted after the end of the segment 1.

That is, in segment 0 of the first row, normal sectors LBA (0), LBA (m+1), LBA (2), LBA (m+3), . . . LBA (m−2), LBA (2m−1), and redundant sector 0 are allocated. In segment 1 of the second row, normal sectors LBA (m), LBA (1), LBA (m+2), LBA (3), . . . LBA (2m−2), LBA (m−1), and redundant sector 1 are allocated.

Operation of the HDD according to the present embodiment will next be described.

The redundant sector data update processing is performed in the same manner as the first embodiment.

The operation of the HDD in the present embodiment differs from that of the first embodiment in the specification of the sector allocation performed in step S22. The MPU 8 specifies, as the sector allocation, the order of the normal sectors that have been alternately allocated between the two segments as described above.

A use of the sector allocation according to the present embodiment enables the redundant sector data generation processing and error sector data recovery processing to be performed in the same manner as the first embodiment and allows required normal sectors to be read out.

According to the present embodiment, in the case where two consecutive error sectors exist in one segment, the error sectors exist one by one in two segments. Thus, by performing the error sector data recovery processing for each segment, the two consecutive error sectors can be recovered.

## Fourth Embodiment

In a fourth embodiment of the present invention, an HDD capable of recovering data of the error sector even in the case where two consecutive error sectors exist in one segment and capable of reducing processing time will be described.

The HDD according to the present embodiment has the same configuration as that according to the first embodiment except for the configuration of the redundant calculation section 21. That is, the calculation result memory 31 has a capacity corresponding to two sectors, and XOR calculation section 32 performs XOR calculation for each bit position using externally input data corresponding to two sectors and calculation result data corresponding to two sectors which are retained in the calculation result memory 31.

The sector allocation in the user area on the disk medium 15 will next be described.

The sector allocation in the present embodiment is the same as that in the third embodiment. For example, as shown in FIG. 10, in segment 0 of the first row, normal sectors LBA (0), LBA (m+1), LBA (2), LBA (m+3), . . . LBA (m−2), LBA (2m−1), and redundant sector 0 are allocated. In segment 1 of the second row, normal sectors LBA (m), LBA (1), LBA (m+2), LBA (3), . . . LBA (2m−2), LBA (m−1), and redundant sector 1 are allocated.

In the redundant sector data generation processing and error sector data recovery processing according to the present embodiment, the MPU 8 reads out two sectors at a time according to the sector allocation, and the redundant calculation section 21 performs XOR calculation for two sectors at a time. In this example, specifically, two sectors of LBA (0) and LBA (1) are read out at a time and input to the redundant calculation section 21. Subsequently, two sectors of LBA (2) and LBA (3), two sectors of LBA (m−2) and LBA (m−1) are read out respectively and are then input to the redundant calculation section 21. Eventually, redundant sector data corresponding to two sectors or error sector data corresponding to two sectors are retained in the calculation result memory 31.

According to the present embodiment, in the case where two consecutive error sectors exist in one segment, the error sectors exist one by one in two segments. Thus, by performing the error sector data recovery processing for each segment, the two consecutive error sectors can be recovered. Further, by using the calculation result memory 31 and XOR calculation section 32 that process data corresponding to two sectors, it is possible to reduce the time required for the redundant sector data generation processing or error sector data recovery processing by about ½ of the time required in the third embodiment.

Although the data length of each of the calculation result memory 31 and XOR calculation section 32 corresponds to data of two sectors in the present embodiment, other data lengths may be employed. Further, although normal sectors having consecutive LBAs are alternately allocated between two segments in the present embodiment, the normal sectors having consecutive LBAs may be allocated between two or more segments according to a predetermined rule.

## Fifth Embodiment

In a fifth embodiment of the present invention, an HDD capable of reducing the load of the data buffer or MPU will be described.

A configuration of the HDD according to the present embodiment will first be described.

FIG. 11 is a block diagram showing an example of a configuration of the HDD according to the present embodiment. In FIG. 11, the same reference numerals denote the same or corresponding parts as in FIG. 1, and the descriptions thereof are omitted here. A different point from FIG. 1 is that a path from the redundant calculation section 21 to the format controller 5 is newly provided.

Operation of the HDD according to the present embodiment will next be described.

Although the redundant sector data update processing is the same as that of the first embodiment, single sector data write processing differs from that of the first embodiment. Other operations are the same as those of the first embodiment.

FIG. 12 is a flowchart showing an example of the single redundant sector data write processing according to the present embodiment. The MPU 8 makes the setting of a data path for the format controller 5 so as to allow the data of the calculation result memory 31 to be directly written from the redundant calculation section 21 onto the disk medium 15 (S33). Then, the MPU 8 writes the redundant sector data in the calculation result memory 31 onto the disk medium 15 in the same manner as the procedure in which the user data in the data buffer 4 are written onto the disk medium 15 (S34) and ends this flow.

As described above, the redundant sector data retained in the calculation result memory 31 of the redundant calculation section 21 are directly written onto the disk medium 15 via the format controller 5. With this configuration, it is possible to reduce the load of the data buffer 4 or MPU 8 (firmware) as compared to the first embodiment.

## Sixth Embodiment

In a sixth embodiment of the present invention, an HDD that adds a redundant sector to a part of the range in the user area will be described.

A set of the normal sectors in the user area to which the redundant sector is to be added is set as a recovery target range, and the normal sectors in the recovery target range are divided into segments. Therefore, only when the normal sectors in the recovery target range includes any error sector, the error sector data recovery processing is performed.

The HDD according to the present embodiment has the same configuration as that according to the first embodiment.

Operation of the HDD according to the present embodiment will next be described.

Recovery target range identification processing which is performed at the power-on time in the case where the recovery target range has been set will first be described.

FIG. **13** is a flowchart showing an example of operation of the recovery target range identification processing according to the present embodiment. The MPU **8** activates the SPM **13** (S**81**). Then the MPU **8** reads out control information required for control of the HDD from the system area of the disk (S**82**) and makes the setting of the HDD based on the read out control information (S**83**). Then the MPU **8** reads out from the system area the first and last LBAs of the recovery target range (S**84**), makes the setting of the recovery target range based on the read out information (S**85**), and ends this flow.

By making the abovementioned setting of the recovery target range, the update range in the redundant sector data update processing is set in the recovery target range. Other operations are the same as those of the first embodiment.

According to the present embodiment, the redundant sector is added only to the normal sector that needs to be recovered, thereby suppressing the consumption of resources such as the memory **9** in the HDD or system area on the disk medium **15**.

## Seventh Embodiment

In a seventh embodiment of the present invention, an HDD capable of changing the size of each segment will be described.

A segment size, which is the size of each segment, stays constant in the abovementioned embodiments; while in the present embodiment, the segment size can be changed such that, for example, the segment size is made smaller in a frequently-accessed area while the segment size is made larger in an infrequently-accessed area.

The HDD according to the present embodiment has the same configuration as that according to the first embodiment.

Operation of the HDD according to the present embodiment will next be described.

The recovery target range identification processing which is performed at the power-on time in the case where the recovery target range has been set will be described.

Segment size identification processing which is performed at the power-on time in the case where the segment size has been set will first be described. FIG. **14** is a flowchart showing an example of operation of the segment size identification processing according to the present embodiment. In this flowchart, processing of steps **81**, **82**, **83** are the same as those in the sixth embodiment. After the step S**83**, the MPU **8** reads out, from the system area, segment information including the segment size (S**86**), makes the setting of the size of each segment and the like based on the read out information (S**87**), and ends this flow.

The segment information stored in the system area on the disk medium **15** will here be described. FIG. **15** is a table showing an example of the segment information according to the present embodiment. The segment information has, as its items, "segment start LBA", "presence/absence of redundant sector data", and "segment size" for each segment. In this example, the area from LBA (0x00000000) to LBA (0x0000ffff) is assigned to segments each having a segment size of 0x0800 sectors, and redundant sectors are assigned to the respective segments. The area from LBA (0x00010000) to LBA (0x0001ffff) is assigned to segments each having a segment size of 0x10000 sectors, and redundant sectors are assigned to the respective segments. On the other hand, no segment and no redundant sector is assigned to the area from LBA (0x00020000) to the last sector.

By making the abovementioned setting of the recovery target range, the update range in the redundant sector data update processing is set in an area to which the redundant sector data has been assigned. Other operations are the same as those of the first embodiment.

By retaining such segment information in the system area, assignation of the segment or redundant sector can be changed for each use of the HDD or for each user of the HDD. Further, when the ratio of the redundant sector data is made higher in a user area where update is frequent, that is, an area where there is a high possibility of occurrence of the error sector, resources of the HDD can effectively be used and possibility that the data can be recovered is significantly increased, thereby considerably increasing the reliability of the HDD.

## Eighth Embodiment

In an eighth embodiment of the present invention, an HDD that has information indicating whether each redundant sector data is valid or not will be described.

The HDD according to the present embodiment has the same configuration as that of the first embodiment.

A redundant sector table indicating whether each redundant sector data is valid or not will next be described.

The redundant sector data table is generated by the MPU **8** and is stored in the non-volatile memory **10**. FIG. **16** is a table showing an example of the data structure of the redundant sector data table according to the present embodiment. The redundant sector data table has, as its items, "redundant sector number (segment number)" corresponding to each redundant sector data and "valid flag" indicating whether each redundant sector data is valid or not. The MPU **8** can update only invalid redundant sector data by referring to the redundant sector data table at the data update time, thereby increasing the efficiency of the update processing of the redundant sector data.

It is preferable that the update of the redundant sector data table be performed while the MPU **8** does not receive a command from the higher-level device.

Operation of the HDD according to the present embodiment will next be described.

Although the redundant sector data update processing according to the present embodiment is the same as that according to the first embodiment, only different point is that processing that searches the redundant sector data table for a segment in which the valid flag indicates "invalid" and sets the found segment as an update range is performed in place of step S**12** of FIG. **4**. Other processing are the same as those of the first embodiment.

Command reception processing performed when the MPU **8** receives a command from the higher-level device will next be described.

FIG. **17** is a flowchart showing an example of operation of the command reception processing according to the present embodiment. The MPU **8** determines whether it has received

any command from the higher-level device. If the MPU **8** has not received any command (N in S**51**), it returns to S**51**, while the MPU **8** has received any command (Y in S**51**), it advances to the next step, where the MPU **8** determines whether the received command is a data write command. If the command is not a data write command (N in S**52**), the MPU **8** advances to S**54**, while if the command is a data write command (Y in S**52**), the MPU **8** advances to the next step, where the MPU **8** performs flag clear processing (S**53**). Then, the MPU **8** performs command processing which is normal processing performed based on the received command (S**54**) and ends this flow.

The flag clear processing that clears (invalidates) a valid flag will next be described.

FIG. **18** is a flowchart showing an example of operation of the flag clear processing according to the present embodiment. The MPU **8** specifies a write target segment which is a target of write processing based on received LBA and write size (S**62**). For example, assuming that the size of all segments is constant, the MPU **8** sets [received LBA/segment size] as the start segment of the write target segment and sets [[received LBA+write size]/segment size] as the end segment thereof. If the segment size is not constant, case analysis is performed using the segment size. Then, the MPU **8** clears the valid flag corresponding to the write target segment (S**63**) and ends this flow.

Assume that the flag clear processing is performed after the data write processing. In this case, if an unexpected trouble occurs between the write processing time and flag clear processing time to cause the valid flag to remain, improper error sector data recovery processing may be performed. Therefore, in the present embodiment, the flag clear processing is performed before the data write processing.

Flag set processing that sets (validates) the valid flag will next be described.

The MPU **8** performs the flag set processing after the completion of the single redundant sector data write processing. In this processing, the MPU **8** sets in the redundant sector data table a valid flag corresponding to the redundant sector data written through the single redundant sector data write processing.

Assume that the flag set processing is performed before the single redundant sector data write processing. In this case, if only the valid flag is set in a state where the redundant sector data has not been written, improper error sector data recovery processing may be performed.

The error sector data recovery processing according to the present embodiment will next be described.

FIG. **19** is a flowchart showing an example of operation of the error sector data recovery processing according to the present embodiment. The MPU **8** specifies a segment to which an error sector belongs based on the detected LBA of the error sector and acquires the valid flag corresponding to the specified segment by referring to the redundant sector data table (S**41**). Then, the MPU **8** determines whether the acquired valid flag is valid. If the valid flag is invalid (N in S**42**), the MPU **8** ends this flow, while if the valid flag is valid (Y in S**42**), the MPU **8** advances to the next step. The subsequent steps S**44**, S**45**, S**46**, S**47**, and S**48** are the same as those in the error sector data recovery processing of the first embodiment.

If the redundant sector data table disappears after the power-off of the HDD, the error sector data recovery processing cannot be performed in the case where there detected any error sector immediately after the next power-on. Thus, in the present embodiment, the redundant sector data table is stored in the non-volatile memory **10**.

The redundant sector data table may be updated and written onto the disk medium **15** at a timing at which the HDD receives a Flush Cache command or a head retreat command from the higher-level device. Since a write cache function is used in HDDs in general, write data from the higher-level device which is retained in the data buffer **4** (cache) is written onto the disk medium **15** before the power-off. The Flush Cache command is used to force the write processing to the disk medium **15** to be performed and is issued from the higher-level device before the power-off of the HDD. At this timing, the MPU **8** performs the update of the redundant sector data table and data write onto the disk medium **15**.

According to the present embodiment, it is possible to update only the redundant sector data that needs to be updated, so that the load associated with the update processing of the redundant sector data can be reduced. Therefore, it is possible to quickly complete the update processing of the redundant sector data while preventing the performance of the HDD from being deteriorated. Further, a possibility that the redundant sector data corresponding to the error sector has already been generated in the case where the error sector has been detected can be increased.

While a possibility that the redundant sector data corresponding to the error sector is valid is increased when the size of all segments is reduced, the consumption of resources such as the memory **9** in the HDD or system area on the disk medium **15** is increased. In order to cope with this problem, the size of each segment is made different from one another like the seventh embodiment, thereby increasing the reliability of the HDD while suppressing the resource consumption.

## Ninth Embodiment

In a ninth embodiment of the present invention, an HDD that dynamically changes the segment information will be described.

The HDD according to the present embodiment has the same configuration as that according to the first embodiment.

Operation of the HDD according to the present embodiment will next be described.

The redundant sector data update processing, redundant sector data generation processing, single redundant sector data write processing, user data read processing, and disk medium scan processing are the same as those of the first embodiment. As in the case of the eighth embodiment, the redundant sector data table is stored in the non-volatile memory **10**. Further, as in the case of the second embodiment, in the sector allocation, all redundant sectors are consecutively allocated in a consecutive manner, following after all normal sectors consecutively allocated. Further, as in the case of seventh embodiment, the segment information is stored in the system area on the disk medium **15**.

Segment information change processing that dynamically changes the segment information will next be described.

FIG. **20** is a flowchart showing an example of operation of the segment information change processing according to the present embodiment. The MPU **8** determines whether it has received any write command from the higher-level device. If the MPU has not received the write command (N in S**91**), it returns to S**91**, while if the MPU **8** has received the write command (Y in S**91**), it advances to the next step, where the MPU **8** calculates the last LBA in the write range (S**92**). Then the MPU **8** determines whether the data has been written in an area for which data write has never been performed. If the data has been written in an area other than the area for which data write has never been performed (N in S**93**), the MPU **8** ends this flow, while if the data has been written in an area for

which data write has never been performed (Y in S**93**), the MPU advances to the next step. The MPU **8** then stores therein the last LBA in the write range as the maximum write LBA (S**94**). After that, the MPU **8** determines whether a predetermined time has elapsed. If a predetermined time has not elapsed (N in S**95**), the MPU ends this flow, while if a predetermined time has elapsed (Y in S**95**), the MPU **8** advances to the next step, where the MPU **8** determines whether the maximum write LBA is larger than a segment to which the redundant sector has been assigned. If the maximum write LBA is not larger then the segment (N in S**96**), the MPU **8** ends this flow, while if the maximum write LBA is larger then the segment (Y in S**96**), the MPU **8** advances to the next step, where the MPU **8** changes the segment information such that the redundant sector is assigned up to the segment including the maximum write LBA (S**97**). Then the MPU **8** initializes the redundant sector valid table to invalid all redundant sectors once (S**98**) and ends this flow.

It takes a long time until the entire disk capacity has been used from the start of the use of an HDD. If all segment information have been set at the time of start of use of the HDD, a redundant sector area corresponding to an unused part of the user area is not used for a while. According to the present embodiment, dynamically changing the segment information can reduce the unused part of the redundant sector area. For example, in the case where the segment is formed, including 0x10000 sectors, over the entire user area, the size of the segment is reduced to 0x08000 in a state where only less than half of the entire user area is used. By doing this, it is possible to effectively use the redundant sector area.

A first block corresponds to the normal sector in the embodiments. A second block corresponds to the segment in the embodiments. A third block corresponds to the redundant sector in the embodiments. A storage medium corresponds to the disk medium in the embodiments. An update target corresponds to the update range in the embodiments. Calculation data corresponds to the calculation result data in the embodiments. A calculation section corresponds to the redundant calculation section in the embodiments. A write section and a read section correspond to the MPU, data buffer, buffer controller, and format controller in the embodiments. A calculation step corresponds to step S**13** in the embodiments. A write step corresponds to step S**14** or step S**19** in the embodiments.

A management information storage section corresponds to the redundant sector data table in the embodiments. A management information storage step corresponds to the flag set processing and flag clear processing in the embodiments. A read step corresponds to the error sector data recovery processing in the embodiments.

Further, it is possible to provide a program that allows a computer constituting a stored data processing apparatus to execute the above steps as a stored data processing program. By storing the above program in a computer-readable storage medium, it is possible to allow the computer constituting the stored data processing apparatus to execute the program. The computer-readable medium mentioned here includes: an internal storage device mounted in a computer, such as ROM (Read Only Memory) or RAM (Random Access Memory), a portable storage medium such as a CD (Compact Disk)-ROM, a flexible disk, a DVD (Digital Versatile Disk) disk, a magneto-optical disk, or an IC (Integrated Circuit) card; a database that holds computer program; another computer and database thereof; and a transmission medium on a network line.

The MPU **8** and redundant calculation section **21** according to the present embodiments can easily be applied to a storage apparatus to increase the performance of the storage apparatus. Examples of the storage apparatus include a magnetic disk apparatus, an optical disk apparatus, a magneto-optical disk apparatus.

What is claimed is:

**1**. A stored data processing apparatus for use in a storage apparatus that performs at least write processing for a storage medium on which data is allocated in units of a first block having a predetermined data length, the first block is allocated in units of a second block constituted by a plurality of the first blocks, and a third block having the same data length as the first block is allocated in correspondence with the second block, comprising:

a management information storage section that stores, for each second block, management information including information indicating whether the third block corresponding to the second block is valid or invalid;

a calculation section that performs calculation for each bit position using data of all the first blocks in an update target in the case where the management information stored in the management information storage section indicates that the third block corresponding to the second block specified as the update area is invalid and outputs a result of the calculation as calculation data;

a write section that writes the calculation data output from the calculation section in the third block associated with the update target and validates the third block in the management information stored in the management information storage section; and

a read section that reads out data from the first block that has been specified as a read target, sets the first block from which data has not normally been read out as an error block in the case where the data read has failed, reads out normal data which is data of all the first blocks other than the error block in the second block in the case where the management information stored in the management information storage section indicates that the third block corresponding to the second block including the read target is valid, and reads out redundant data which is data of the third block corresponding to the second block,

wherein the calculation section further performs calculation for each bit position using the normal data and redundant data and outputs a result of the calculation as data of the error block.

**2**. The stored data processing apparatus according to claim **1**, wherein the calculation section initializes the calculation data, sequentially sets data of the respective normal data and redundant data as input data of the calculation, performs calculation for each bit position using the input data and calculation data, stores a result of the calculation as new calculation data, and repeats the calculation and storage for all the input data to output the calculation data.

**3**. The stored data processing apparatus according to claim **1**, wherein when receiving a data write instruction from an external device, the write section validates the third block corresponding to the second block including the data write target specified by the data write instruction in the management information stored in the management information storage section.

**4**. The stored data processing apparatus according to claim **1**, wherein the calculation section performs the calculation for the second block corresponding to the third block which is invalidated in the management information stored in the management information storage section while it does not perform processing based on a data read instruction or data write instruction from an external device.

**5**. The stored data processing apparatus according to claim **1**, wherein the management information storage section is a non-volatile memory.

6. The stored data processing apparatus according to claim 1, wherein the management information storage section writes management information onto the storage medium at a timing of receiving, from an external device, an instruction of writing write cache data onto the storage medium or instruction of retreating a head.

7. The stored data processing apparatus according to claim 1, wherein the management information includes specification of a plurality of first blocks included in the second block, which is made for each second block.

8. The stored data processing apparatus according to claim 1, wherein:

the management information includes specification of a plurality of first blocks included in the second block, which is made for each second block, and

the write section allocates the second block on the storage medium based on the management information.

9. The stored data processing apparatus according to claim 8, wherein the management information storage section determines the number of first blocks included in the second block based on the access frequency from an external device to each second block so as to set the determined number in the management information.

10. The stored data processing apparatus according to claim 1, wherein the calculation performed by the calculation section is a calculation in which the same calculation result can be obtained even if the order of the calculation is changed.

11. The stored data processing apparatus according to claim 10, wherein the calculation performed by the calculation section is exclusive OR.

12. The stored data processing apparatus according to claim 1, wherein:

the first block is at least one sector,

the third block has the same data length as the first block, and

the calculation data has the same data length as the first block.

13. The stored data processing apparatus according to claim 1, wherein when the calculation section outputs the data of the error block, the write section performs reassignment processing to assign an empty block to the error block.

14. A stored data processing apparatus for use in a storage apparatus that performs at least write processing for a storage medium on which data is allocated in units of a first block having a predetermined data length, the first block is allocated in units of a second block constituted by a plurality of the first blocks, and a third block having the same data length as the first block is allocated in correspondence with the second block, comprising:

a management information storage section that stores, for each second block, management information including information indicating whether the third block corresponding to the second block is valid or invalid;

a calculation section that performs calculation for each bit position using data of all the first blocks in an update target in the case where the management information stored in the management information storage section indicates that the third block corresponding to the second block specified as the update area is invalid and outputs a result of the calculation as calculation data; and

a write section that writes the calculation data output from the calculation section in the third block associated with the update target and validates the third block in the management information stored in the management information storage section,

wherein the calculation section initializes the calculation data, sequentially sets data of the respective first blocks in the update target as input data of the calculation, performs calculation for each bit position using the input data and calculation data, stores a result of the calculation as new calculation data, and repeats the calculation and storage for all the input data to output the calculation data.

15. A storage apparatus that performs at least write processing for a storage medium on which data is allocated in units of a first block having a predetermined data length, the first block is allocated in units of a second block constituted by a plurality of the first blocks, and a third block having the same data length as the first block is allocated in correspondence with the second block, comprising:

a management information storage section that stores, for each second block, management information including information indicating whether the third block corresponding to the second block is valid or invalid;

a calculation section that performs calculation for each bit position using data of all the first blocks in an update target in the case where the management information stored in the management information storage section indicates that the third block corresponding to the second block specified as the update area is invalid and outputs a result of the calculation as calculation data;

a write section that writes the calculation data output from the calculation section in the third block associated with the update target and validates the third block in the management information stored in the management information storage section,

a read section that reads out data from the first block that has been specified as a read target, sets the first block from which data has not normally been read out as an error block in the case where the data read has failed, reads out normal data which is data of all the first blocks other than the error block in the second block in the case where the management information stored in the management information storage section indicates that the third block corresponding to the second block including the read target is valid, and reads out redundant data which is data of the third block corresponding to the second block,

wherein the calculation section further performs calculation for each bit position using the normal data and redundant data and outputs a result of the calculation as data of the error block.

16. The storage apparatus according to claim 15, wherein the calculation section initializes the calculation data, sequentially sets data of the respective first blocks in the update target as input data of the calculation, performs calculation for each bit position using the input data and calculation data, stores a result of the calculation as new calculation data, and repeats the calculation and storage for all the input data to output the calculation data.

17. The storage apparatus according to claim 15, wherein the calculation section initializes the calculation data, sequentially sets data of the respective normal data and redundant data as input data of the calculation, performs calculation for each bit position using the input data and calculation data, stores a result of the calculation as new calculation data, and repeats the calculation and storage for all the input data to output the calculation data.

18. The storage apparatus according to claim 15, wherein when receiving a data write instruction from an external device, the write section validates the third block corresponding to the second block including the data write target specified by the data write instruction in the management information stored in the management information storage section.

* * * * *